Sharable Content Object Reference Model (SCORM™) Version 1.1

Conformance Requirements and Certification Testing Procedures

Version 1.1.1

August 30, 2001

This page intentionally left blank.

Advanced Distributed Learning
Sharable Content Object Reference Model
Conformance Requirements and Certification Testing Procedures
Version 1.1.1

Available at ADLNet
(http://www.adlnet.org)

For questions and comments visit the ADL Help & Info Center at http://www.adlnet.org.

This page intentionally left blank.

Table of Contents

This page intentionally left blank.

1. SECTION I (Page Number Style)

SECTION 1

Introduction

This page intentionally left blank.

1.1. Purpose

The Department of Defense (DoD) established the Advanced Distributed Learning (ADL) initiative to develop a DoD-wide strategy for using learning and information technologies to modernize education and training. In order to leverage existing practices, promote the use of technology-based learning and provide a sound economic basis for investment, the ADL initiative has defined high level requirements for learning content such as content reusability, accessibility, durability and interoperability.

The Sharable Content Object Reference Model (SCORM™) Version 1.1[15], released in January 2001, defines a reference model for sharable learning content objects that meet ADL high level requirements. The SCORM is an integrated collection of technical specifications that enable conforming Web-based learning products and learning content to interoperate.

It is highly recommended that the reader be familiar with the SCORM Version 1.1 before reading this document.

1.2. Scope

The Advanced Distributed Learning (ADL) initiative provides an overall certification process of which this document is one part. This document defines the requirements put forth by the SCORM Version 1.1 that must be implemented by Learning Management Systems (LMSs) and/or Learning Content in order to attain certification of conformance with the SCORM Version 1.1. In addition, this document also defines the detailed procedures that must be followed to technically assess and certify the conformance of LMSs and Learning Content to the SCORM Version 1.1.

At this time, ADL will not offer certification for Content Structure Format (CSF) XML instance documents. The SCORM Version 1.2 will map the CSF elements into the IMS Content Packaging specification[8] to form the SCORM Content Aggregation Packaging Profile. ADL will incorporate the capability to test Content Packages in conjunction with this effort. In this vein, the LMS CSF import is not required of LMSs for certification. The LMS CSF import requirement will be superceded by a requirement that LMSs be able to import Content Packages upon release of the SCORM Version 1.2.

It is important to note that content aggregations (e.g. entire courses and/or portions of courses) will not be tested for conformance at this time. Conformance testing of individual SCOs and meta-data documents is necessary at this time because the SCORM Version 1.1 does not address content packaging. Upon release of the SCORM Version 1.2, which will address content packaging for aggregated content, it is anticipated that certification testing for conformance of aggregations of SCOs and associated Meta-data will be offered.

This document does not cover the testing of stand-alone Content Structure Format XML instance documents, Authoring Tools, or Repositories.

1.3. Conformance Requirements Overview

This section contains three tables that provide a high-level summary of the test subjects that can be certified for conformance to the SCORM Version 1.1. These tables list the conformance labels that are used to describe test subjects that are verified to be conformant to a particular feature or capability described in the SCORM along with a brief description of the requirements that must be implemented by the test subject in order to achieve the corresponding conformance label.

The detailed requirements for conformance are specified in Section 2. The procedures for using the ADL SCORM Conformance Test Suite Version 1.1.1 to verify conformance of test subjects is contained in Section 3.

Learning Management System (LMS) Conformance Categories
SCORM Version 1.1 Run-Time Environment Conformant – Minimum *Conformance Label*: **LMS–RTE1** *Requirements Summary:* The LMS: • Is able to launch a known conformant Sharable Content Object (SCO) as defined in Section 3.2 of the SCORM Version 1.1, and • Provides and exposes an API Adapter as a Document Object Model (DOM) object and correctly implements all of the API functions as described in the SCORM Version 1.1 Section 3.3, and • Correctly Implements support for all required SCORM Version 1.1 Run-time Environment Data Model Mandatory Elements as described in Section 3.4 of the SCORM Version 1.1. *Note: If the LMS **incorrectly** implements one or more SCORM Version 1.1 Run-time Environment Data Model Optional Elements, and does not implement any other optional data model elements correctly, then the LMS is still considered to be LMS-RTE1 conformant as long as the criteria above are met.*
SCORM Version 1.1 Run-Time Environment Conformant - Minimum with Some Optional Data Model Elements *Conformance Label*: **LMS–RTE2** *Requirements Summary:* The LMS: • Is "SCORM Version 1.1Run-Time Environment Conformant – Minimum" and • Correctly implements one or more (not all) of the optional SCORM Version 1.1 Run-time Environment Data Model Optional Elements. *Note: If the LMS **incorrectly** implements one or more SCORM Version 1.1 Run-time Environment Data Model Optional Elements, the LMS can still be considered LMS-RTE2 conformant as long as one or more other data model elements are implemented correctly.*
SCORM Version 1.1 Run-Time Environment Conformant - Minimum with All Optional Data Model Elements *Conformance Label*: **LMS–RTE3** *Requirements Summary:* The LMS: • Is "SCORM Version 1.1 Run-Time Environment Conformant – Minimum", and • Correctly implements **all** of the optional SCORM Version 1.1 Run-Time Environment Data Model Optional Elements.

Table 1.3.3a – Learning Management System Run-Time Environment Conformance Matrix

Sharable Content Object (SCO)
Conformance Categories

SCORM Version 1.1 Run-Time Environment Conformant – Minimum

Conformance Label: **SCO–RTE1**

Requirements Summary: The SCO:

- Is able to be launched by a known conformant LMS as defined in Section 3.2 of the SCORM Version 1.1, and
- Searches for and finds an API Adapter as a Document Object Model (DOM) object, and
- Implements, at a minimum, the LMSInitialize() and LMSFinish() API functions as described in the SCORM Version 1.1 Section 3.3, and
- Any additional API functions that are called are called correctly.

SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Mandatory Data Model Elements

Conformance Label: **SCO–RTE1+Mandatory**

Requirements Summary: The SCO:

- Is "SCORM Version 1.1 Run-Time Environment Conformant – Minimum", and
- Implements support for correctly getting and/or setting one or more LMS **mandatory** SCORM Version 1.1 Run-time Environment Data Model Elements. (Note LMS mandatory is defined as those data model elements that are required to be implemented by an LMS.)

*Note: If the SCO **incorrectly** implements one or more mandatory SCORM Version 1.1 Run-time Environment Data Model Elements, the SCO is non-conformant*

SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Optional Data Model Elements

Conformance Label: **SCO–RTE1+Optional**

Requirements Summary: The SCO:

- Is "SCORM Version 1.1 Run-Time Environment Conformant – Minimum", and
- Implements support for correctly getting and/or setting one or more LMS **optional** SCORM Version 1.1 Run-time Environment Data Model Elements. (Note LMS optional is defined as those data model elements that are optional for implementation by an LMS.)

*Note: If the SCO **incorrectly** implements one or more optional SCORM Version 1.1 Run-time Environment Data Model Elements, the SCO is non-conformant.*

Sharable Content Object (SCO) Conformance Categories
SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Optional and Some Mandatory Data Model Elements *Conformance Label*: **SCO–RTE1+Mandatory+Optional** *Requirements Summary:* The SCO is: • "SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Mandatory Data Model Elements", and • "SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Optional Data Model Elements".

Table 1.3.3b – Sharable Content Object Run-Time Environment Conformance Matrix

Course, Content and Raw Media Meta-data Conformance Categories

SCORM Version 1.1 Meta-data XML Conformant – Minimum

Conformance Label: **MD-XML1**

Requirements Summary: The Course, Content, or Raw Media Meta-data XML Instance:

- Is a well formed XML Document, and
- Is valid against the IMS Learning Resource Metadata Version 1.1 XML Document Type Definition (DTD) (Note for Version 1.1, ADL is only testing using the non-name spaced version of the IMS DTD.), and
- Contains elements that conform to their corresponding specified data types, and
- Contains all mandatory document elements for the corresponding category (Course, Content, Raw Media) as described in Section 2.2.3 of the SCORM Version 1.1, and
- Elements defined as having restricted vocabularies adhere to all defined vocabularies as defined in Section 2.2.3 of the SCORM Version 1.1.

SCORM Version 1.1 Meta-data XML Conformant – Minimum with Optional Elements

Conformance Label: **MD-XML1+Optional**

Requirements Summary: The Course, Content, or Raw Media Meta-data XML Instance:

- Is "SCORM Version 1.1 Meta-data XML Conformant – Minimum", and
- Contains one or more elements that are designated as optional meta-data elements for the corresponding meta-data type (Course, Content, Raw Media) as described in Section 2.2.3 of the SCORM Version 1.1, except for extensions (e.g. <extension>).

*Note: If the Meta-data instance **incorrectly** implements one or more elements that are designated as optional document elements for the corresponding meta-data type (Course, Content, Raw Media) as described in Section 2.2.3 of the SCORM Version 1.1 the Meta-data Instance is non-conformant.*

SCORM Version 1.1 Meta-data XML Conformant – Minimum with Extensions

Conformance Label: **MD-XML1+Extensions**

Requirements Summary: The Course, Content, or Raw Media Meta-data XML Instance:

- Is "SCORM Version 1.1 Meta-data XML Conformant – Minimum", and
- Contains one or more extensions. Note: extensions must be defined within the extensions tag (i.e. <extension>) according to the IMS Learning Resource Meta-data Version 1.1. DTD. The SCORM Version 1.1 Conformance Test Suite will ignore elements defined within the <extension> tag when validating against the IMS Learning Resource Meta-data DTD.

Course, Content and Raw Media Meta-data Conformance Categories
SCORM Version 1.1 Meta-data XML Conformant – Minimum with Optional Elements and Extensions *Conformance Label*: **MD-XML1+Optional+Extensions** *Requirements Summary:* The Course, Content, or Raw Media Meta-data XML Instance: • Is "SCORM Version 1.1 Meta-data XML Conformant – Minimum with Optional Elements", and • Is "SCORM Version 1.1 Meta-data XML Conformant – Minimum with Extensions".

Table 1.3.3c – Meta-Data XML Conformance Matrix

1.3.1. Learning Management System Test Overview

The purpose of the LMS Run-Time Environment Test is to verify that a test subject LMS implements the conformance requirements as outlined in Section 2. The LMS Run-Time Environment Test uses a set of two test courses that are to be loaded or imported into the LMS. The Sharable Content Objects (SCOs) that comprise the test course(s) exercise the various aspects of the LMSs implementation of the SCORM Version 1.1 Run-Time Environment. Based on the LMS Run-Time Environment Conformance Requirements, the LMS may or may not be found to be SCORM Version 1.1 Conformant in terms of the conformance categories that are outlined in Table 1.3.3a above.

The ability to import a course, whether automatically, using a CSF, or through some manual process, is a prerequisite for performing the LMS Run-Time Environment Test. It is not necessary that the LMS have the ability to do this using a CSF in order to attain certification to any of the LMS Run-Time Environment conformance categories.

1.3.2. Sharable Content Object Test Overview

The purpose of the SCO Run-Time Environment Test is to verify that a test subject SCO implements the conformance requirements as outlined in Section 2. The SCO Run-Time Environment Test software simulates an LMS that is "SCORM Version 1.1 Run-Time Environment Conformant – Minimum with All Optional Data Model Elements" (LMS-RTE3). The test subject SCO is launched by the test software and is expected to search for and find the LMSs API Adapter. The test software then services and audits any API function calls that are made by the SCO. As the Test Suite operator executes the SCOs functionality (simulates the learner experience), the test software audits the SCORM Version 1.1 Run-Time Environment Data Model elements that are implemented by the SCO.

The conformance category assigned to the SCO is dependent upon the SCOs ability to perform the minimum API interaction with the LMS and then, additionally, whether the SCO implements any mandatory or optional data model elements, where mandatory or optional are defined in terms of the LMSs obligation for implementation (i.e. LMS mandatory or LMS optional). The SCO may or may not be found to be SCORM Version 1.1 Conformant in terms of the conformance categories that are outlined in Table 1.3.3b above.

It is important to recognize that certain SCOs may be very simple in nature, offering a single logical path of execution from start to finish. Other SCOs may be more complex, having several possible paths of execution that are conditional, based on user performance or user personalization variables, for example. For this reason it is not possible to programmatically test all conditional branches of the SCOs using a "black box" testing approach. Furthermore, without placing specific constraints on implementation technologies, it is not feasible to inspect the implementation details of the

SCO to validate conformance, as would be required using a "white box" testing approach.

ADL has chosen a subjective approach that involves a compromise between the "black box" and "white box" approaches. When testing a SCO that contains conditional logic that involves the potential for the SCO to interact with an LMS using data model element sets that vary across the conditional paths, or to execute varying sets of API functions, the path for the test is left to the discretion of the Test Suite operator. The Test Suite operator may, if testing multiple SCOs from a single organization, elect to employ different strategies across multiple SCOs, or even within one single SCO.

For this reason, the SCO Run-Time Environment Test does not guarantee that the SCO correctly implements the SCORM Version 1.1 Run-Time Environment in all cases, but only within the bounds of the scenario that was used for an instance of testing.

It is also important to realize that the SCO Run-Time Environment Test does not guarantee that the SCO is without defects. The test software does not validate that all aspects of the SCO implementation are accurate and/or correct. For example, it is quite possible that a SCO can be found SCORM Version 1.1 Conformant within one of the previously mentioned categories and still contain run-time defects (e.g. broken links, JavaScript runtime errors, etc.), and/or not be instructionally sound.

1.3.3. Meta-data Test Overview

The purpose of the Meta-data test is to determine if a test subject Raw Media, Content or Course Meta-data instance is SCORM Version 1.1 Conformant in terms of the conformance categories that are outlined in Table 1.3.3c above. The test checks the Meta-data instance to determine if it:

1. Validates against the IMS Learning Resource Meta-data DTD Version 1.1[10]. Please note that only the non-name spaced version of the IMS Learning Resource Meta-data DTD is used for testing.
2. Implements all mandatory elements
3. Adheres to specified data types and restricted vocabularies
4. Does or does not implement any optional elements
5. Implements any optional elements correctly
6. Does or does not implement extensions (i.e. <extension>).

1.4. Conformance Verification Methodology

Where possible, conformance requirements are validated in an automated fashion, using the SCORM Version 1.1 Conformance Test Suite Version 1.1.1. It is not feasible to validate the implementation of all detailed conformance requirements using the automated Test Suite. Some requirements are validated through manual inspection or observation by the Test Suite operator (auditor). The detailed conformance testing procedures in Section 3 defines the method by which the implementation of the conformance requirements are validated.

2. SECTION II (Page Number Style)

SECTION 2

Conformance Requirements

This page intentionally left blank.

This section describes the detailed requirements that must be implemented by a test subject in order to attain conformance with the SCORM Version 1.1. Each of the requirements contained in this section will be verified using the conformance test procedures contained in Section 3. Where possible, the requirements are validated automatically using the conformance test software. This is not feasible for all requirements; therefore some requirements must be validated manually, by inspection or through observation at runtime. The verification method for all requirements contained in the following sections is automatic, using the test software, unless otherwise noted in the requirements statement.

2.1. LMS Run-Time Environment Conformance Requirements

In order to become certified as SCORM Version 1.1 Run-Time Environment Conformant, an LMS is required to support the Run-time Environment that is described in Section 3 of the SCORM Version 1.1. There are three main components of the SCORM Run-time Environment:

1. Launch
2. Application Program Interface (API)
3. Data Model

The conformance requirements for LMS Run-Time Environment Conformance are broken down into the following sections to address each of the Run-time environment components individually. They are as follows:

- Section 2.1.1 describes the Launch related conformance requirements
- Section 2.1.2 describes the API related conformance requirements
- Section 2.1.3 describes the Data Model related conformance requirements

As described in the Conformance Matrix in Section 1, the LMS Run-Time Environment Conformance requirements are defined in terms of three distinct categories. The three categories are mutually exclusive. An LMS, if conformant, will be designated as conformant within one and only one of these categories. The categories are:

- SCORM Version 1.1 Run-time Environment Conformant – Minimum (LMS–RTE1)
- SCORM Version 1.1 Run-time Environment Conformant – Minimum with Some Optional Data Model Elements (LMS–RTE2)
- SCORM Version 1.1 Run-time Environment Conformant – Minimum with All Optional Data Model Elements (LMS–RTE3)

All three of the above categories of LMS Run-Time Environment Conformance require that the LMS implement all Launch and API requirements described in Section 2.1.1 and Section 2.1.2 respectively.

The conformance requirements for the Run-time Environment Data Model, described in Section 2.1.3, vary across the three conformance categories. The specific requirements for each of the 3 categories of conformance are described in the table in Section 2.1.3.

2.1.1. LMS Run-Time Environment Launch Requirements

It is the responsibility of the LMS to determine which SCO is to be launched. The LMS may launch the SCO automatically in an adaptive fashion or alternatively, may provide a user driven navigation control such as a menu, a table of contents, or back/next buttons. The only requirements specified in this regard, at this time, are that the LMS be able to launch known conformant SCOs, and do so in the order in which the SCOs are defined within the Content Structure Format document (unless prerequisites are defined, which override the sequential ordering).

The LMS shall adhere to the requirements in the following table in order to be considered LMS Run-Time Environment Conformant in any of the three conformance categories (LMS-RTE1, LMS-RTE2, or LMS-RTE3).

Req No	Requirement
2.1.1-1	The LMS shall be able to launch a known SCORM Run-time Environment conformant SCO.
2.1.1-2	The LMS shall launch SCOs using the HTTP protocol.
2.1.1-3	If the LMS supports the ability to import a Content Structure Format (CSF) XML document, then the LMS shall launch SCOs defined in the CSF based on the corresponding SCOs content|block|sco|launch|location element value as defined in the CSF.
2.1.1-4	The LMS shall launch a SCO in a Document Object Model (DOM) frameset child window, or new browser (DOM) window relative to the LMSs controlling browser (DOM) window.
2.1.1-5	The LMS shall launch a SCO such that one and only one SCO is available to the user. There may only be one “currently executing” SCO.

Table 2.1.1a LMS Run-Time Environment Launch Conformance Requirements

2.1.2. LMS Run-Time Environment API Requirements

The SCORM Run-Time Environment API provides a consistent means by which SCOs can communicate and exchange data with LMSs. The LMS is required to implement an API Adapter that supports the functions defined in Section 3 of the SCORM Version 1.1.

The LMS shall adhere to the requirements in the following table in order to be considered LMS Run-Time Environment Conformant in any of the three conformance categories (LMS-RTE1, LMS-RTE2, or LMS-RTE3).

Req No	Requirement
2.1.2-1	The LMS shall expose the required API Adapter as a Document Object Model (DOM) Object named "API" (case sensitive) in the DOM parent or opener window or recursively, in the parent window hierarchy of the parent or opener window of the launched SCO.
2.1.2-2	The LMS shall implement the API Adapter such that it's implementation is accessible (by a SCO) using ECMAScript (JavaScript).
2.1.2-3	The LMS shall provide for the ability for a SCO to call the defined, required API Adapter functions using the defined API Adapter object interface convention (e.g. var cmibooleanResult = API.Function (parameters) using JavaScript.
2.1.2-4	The LMS shall provide implementations of each of the required API Adapter functions, according to the following specified function signatures (parameters and return values): • CMIBoolean = API.LMSInitialize("") • CMIBoolean = API.LMSFinish("") • CMIBoolean = API.LMSCommit(""); • CMIBoolean = API.LMSSetValue(parameter, value) • String = API.LMSGetValue(parameter) • String = API.LMSGetLastError() • String = API.LMSGetErrorString(errorString) • String = API.LMSGetDiagnostic(parameter)
2.1.2-5	The LMS shall provide an implementation for the API Adapter function LMSInitialize
2.1.2-5.1	LMSInitialize shall accept only an empty string parameter ("").

Req No	Requirement
2.1.2-5.2	LMSInitialize shall return a CMIBoolean string value: • A value of "true" indicates that the function completed successfully. • A value of "false" indicates that the function did not complete successfully.
2.1.2-5.3	LMSInitialize shall set the API Error Code to "0" (No Error) if it completes successfully and return "true".
2.1.2-5.4	LMSInitialize shall set the API Error Code to "101" if it does not complete successfully for any reason that is not specified by a more specific or appropriate error code and return "false".
2.1.2-5.5	If LMSInitialize is called with anything other than an empty string ("") parameter, LMSInitialize shall return a value of "false" and set API Error Code "201".
2.1.2-5.6	LMSInitialize shall return false and set the API Error Code to "101" if called more than once by a launched SCO.
2.1.2-6	The LMS shall provide an implementation for the API Adapter function LMSFinish.
2.1.2-6.1	LMSFinish shall accept only an empty string parameter ("").
2.1.2-6.2	LMSFinish shall return a CMIBoolean string value. • A value of "true" shall indicate that the function completed successfully. • A value of "false" shall indicate that the function did not complete successfully. If possible, the LMS should also set an appropriate API Error Code if LMSFinish does not succeed.
2.1.2-6.3	LMSFinish shall set the API Error Code to "0" (No Error) if it completes successfully and return "true".
2.1.2-6.4	LMSFinish shall set the API Error Code to "101" if it does not complete successfully for any reason that is not specified by a more specific or appropriate error code and return "false".
2.1.2-6.5	If LMSFinish is called with anything other than an empty string ("") parameter, LMSFinish shall return a value of "false" and set API Error Code "201".

Req No	Requirement
2.1.2-6.6	The call to LMSFinish by a SCO shall only succeed if the SCO previously called LMSInitialize and the call to LMSInitialize was successful. If the LMS is not initialized, LMSFinish shall fail, returning a value of "false" and setting the API Error Code to "301".
2.1.2-6.7	LMSFinish shall cause any previously set data model elements that were not persisted by the LMS to be persisted. (This applies to LMS implementations that may cache data model element values, as opposed to implementations that persist data model element values at the time of LMSSetValue execution.)
2.1.2-7	The LMS shall provide an implementation for the API Adapter function LMSCommit
2.1.2-7.1	LMSCommit shall accept only an empty string parameter ("").
2.1.2-7.2	LMSCommit shall return a CMIBoolean string value: • A value of "true" shall indicate that the function completed successfully. • A value of "false" shall indicate that the function did not complete successfully. If possible, the LMS should also set an appropriate API Error Code if LMSCommit does not succeed.
2.1.2-7.3	LMSCommit shall set the API Error Code to "0" (No Error) if it completes successfully and return "true".
2.1.2-7.4	LMSCommit shall set the API Error Code to "101" if it does not complete successfully for any reason that is not specified by a more specific or appropriate error code and return "false".
2.1.2-7.5	If LMSCommit is called with anything other than an empty string ("") parameter, LMSCommit shall return a value of "false" and set API Error Code "201".
2.1.2-7.6	The call to LMSCommit by a SCO shall only succeed if the SCO previously called LMSInitialize and the call to LMSInitialize was successful. If the LMS is not initialized, LMSCommit shall fail, returning a value of "false" and setting the API Error Code to "301".
2.1.2-7.7	LMSCommit shall cause any previously set data model elements that have not been persisted to be persisted. (If the LMS caches data model elements set by the SCO using LMSSetValue, the LMSCommit function should persist any currently cached data model values.)

Req No	Requirement
2.1.2-8	The LMS shall provide an implementation for the API Adapter function LMSGetValue
2.1.2-8.1	LMSGetValue shall accept a string parameter containing the name of the data model element requested by the SCO. This is the fully qualified, case sensitive name of the data element as defined in the SCORM Run-time Environment Data Model.
2.1.2-8.2	LMSGetValue shall return a string value.
2.1.2-8.2.1	If the data model element parameter provided by the SCO corresponds to a valid data model element that is readable by the SCO, then LMSGetValue shall return a string containing the appropriate data value. The returned value shall conform to the specifications of its corresponding data type and/or restricted vocabulary list.
2.1.2-8.2.2	If the CMI data model element requested by the SCO does not exist, LMSGetValue shall return an empty string ("") and set API Error Code "201". (Note, API Error Code "401" is also permissible in this case.)
2.1.2-8.2.3	If LMSGetValue is called with a parameter that includes a "._children" extension and the data model element specified does not support _children, then LMSGetValue shall return an empty string ("") and set API Error Code "202". (Note, API Error Code "401" is also permissible in this case.)
2.1.2-8.2.4	If LMSGetValue is called with a parameter that includes a "._count" extension, and the data model element is not an array, then LMSGetValue shall return an empty string ("") and set API Error Code "203". (Note, API Error Code "401" is also permissible in this case.)
2.1.2-8.2.5	If LMSGetValue is called with a parameter that specifies a CMI optional data model element that is not supported by the LMS, LMSGetValue shall return an empty string and set API Error Code "401". (Note, API Error Code "201" is also permissible in this case.)
2.1.2-8.2.6	If LMSGetValue is called with a parameter that specifies a data model element that is not readable by a SCO (write-only by a SCO), LMSGetValue shall return an empty string ("") and set API Error Code "404".
2.1.2-8.3	LMSGetValue shall set the API Error Code to "0" (No Error) if it completes successfully.

Req No	Requirement
2.1.2-8.4	LMSGetValue shall set the API Error Code to "101" if it does not complete successfully for any reason that is not specified by a more specific or appropriate error code.
2.1.2-8.5	The call to LMSGetValue by a SCO shall only succeed if the SCO previously called LMSInitialize and the call to LMSInitialize was successful. If the LMS is not initialized, LMSGetValue shall fail, returning an empty string ("") and setting the API Error Code to "301".
2.1.2-9	The LMS shall provide an implementation for the API Adapter function LMSSetValue.
2.1.2-9.1	LMSSetValue shall accept two string parameters (LMSSetValue(parameter, value))
2.1.2-9.1.1	The first string parameter is the fully qualified, case sensitive string containing the name of the data model element requested to be set by the SCO.
2.1.2-9.1.2	The second parameter is the string containing the value of the data model element that the SCO is requesting to be set.
2.1.2-9.2	LMSSetValue shall return a CMIBoolean string value. • A value of "true" shall indicate that the function completed successfully. • A value of "false" shall indicate that the function did not complete successfully.
2.1.2-9.3	LMSSetValue shall function as follows
2.1.2-9.3.1	If the data model element parameter provided by the SCO corresponds to a valid data model element that is writeable by the SCO, and the value provided by the SCO is of the appropriate specified data type, then LMSSetValue shall set (cache in LMS component/service memory or persist to physical storage) the value for the corresponding data model element and return a string containing "true".
2.1.2-9.3.2	If the data model element parameter provided by the SCO does not exist, LMSSetValue shall return "false" and set API Error Code "201". (Note Error Code "401" is also a permissible error code in this case).
2.1.2-9.3.3	If LMSSetValue is called with a parameter that includes "._children", "._count", or "._version", then LMSSetValue shall return "false" and set API Error Code "402". ". (Note, API Error Code "401" is also permissible in this case.)

Req No	Requirement
2.1.2-9.3.4	If LMSSetValue is called with a parameter that specifies an optional data model element that is not supported by the LMS, LMSSetValue shall return “false” and set API Error Code “401”. (Note, API Error Code “201” is also permissible in this case.)
2.1.2-9.3.5	If LMSSetValue is called with a parameter that specifies a data model element that is not writeable by a SCO (is read-only by a SCO), LMSSetValue shall return “false” and set API Error Code “403”.
2.1.2-9.3.6	If LMSSetValue is called with a data model element value that does not conform to the specified data type for the corresponding data model element, then LMSSetValue shall return “false” and set API Error Code “405”.
2.1.2-9.3.7	If LMSSetValue is called with a data model element that is valid, but is also a list element, the index provided for the element must be less than or equal to a number that is one greater than the current number of elements in the list. If the index is greater than one greater than the current number of elements, then LMSSetValue shall return false and set API Error Code “405”. (List elements are zero based and the list must be sequential. For example, if the “cmi.objectives” list contains 2 elements, an attempt to set cmi.objectives.5.id would result in LMSSetValue returning “false” and setting API Error Code “405”.
2.1.2-9.4	LMSSetValue shall set the API Error Code to “0” (No Error) if it completes successfully and return “true”.
2.1.2-9.5	LMSSetValue shall set the API Error Code to “101” if it does not complete successfully for any reason that is not specified by a more specific or appropriate error code and return “false”.
2.1.2-9.6	The call to LMSSetValue by a SCO shall only succeed if the SCO previously called LMSInitialize and the call to LMSInitialize was successful. If the LMS is not initialized, LMSSetValue shall fail, returning an empty string (“”) and setting the API Error Code to “301”.
2.1.2-10	The LMS shall provide an implementation for the API Adapter function LMSGetLastError
2.1.2-10.1	LMSGetLastError shall accept no parameters.

<table>
<tr><th>Req No</th><th>Requirement</th></tr>
<tr><td>2.1.2-10.2</td><td>LMSGetLastError shall return a string value representing the API Error Code set by the most previously called API Adapter function. The return value shall be one of the following API Error Codes:
“0” (No error)
“101” (General exception)
“201” (Invalid argument error)
“202” (Element cannot have children)
“203” (Element not an array – Cannot have count)
“301” (Not initialized)
“401” (Not implemented error)
“402” (Invalid set value, element is a keyword)
“403” (Element is read only)
“404” (Element is write only)
“405” (Incorrect data type)</td></tr>
<tr><td>2.1.2-10.3</td><td>If no API Adapter functions are called prior to a call to LMSGetLastError (including LMSInitialize), then LMSGetLastError shall return “0” (No error).

Note that the purpose of LMSGetLastError is to return the error code set by the previously called API Adapter function. If there has been no previous successful call to LMSInitialize, then the error code remains at its initial state value, “0”.</td></tr>
<tr><td>2.1.2-11</td><td>The LMS shall provide an implementation for the API Adapter function LMSGetErrorString.</td></tr>
<tr><td>2.1.2-11.1</td><td>LMSGetErrorString shall accept a string parameter containing a valid API Error Code.</td></tr>
<tr><td>2.1.2-11.2</td><td>LMSGetErrorString shall return a string value representing the string description of the API Error Code parameter provided. The return value shall be the API Error Description that corresponds to the provided API Error Code parameter as follows (not case sensitive):

<table>
<tr><th>Error Code Parameter</th><th>Return Value API Error Description String</th></tr>
<tr><td>“0”</td><td>“No error”</td></tr>
<tr><td>“101”</td><td>“General exception”</td></tr>
<tr><td>“201”</td><td>“Invalid argument error”</td></tr>
<tr><td>“202”</td><td>“Element cannot have children”</td></tr>
<tr><td>“203”</td><td>“Element not an array – Cannot have</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Req No</th><th>Requirement</th></tr>
<tr><td></td><td>
<table>
<tr><td></td><td>count”</td></tr>
<tr><td>“301”</td><td>“Not initialized”</td></tr>
<tr><td>“401”</td><td>“Not implemented error”</td></tr>
<tr><td>“402”</td><td>“Invalid set value, element is a keyword”</td></tr>
<tr><td>“403”</td><td>“Element is read only”</td></tr>
<tr><td>“404”</td><td>“Element is write only”</td></tr>
<tr><td>“405”</td><td>“Incorrect Data Type”</td></tr>
</table>
</td></tr>
<tr><td>Note</td><td>The LMS must support the ability for a SCO to call LMSGetErrorString prior to calling LMSInitialize. This is necessary because error state (Error Code) information can be modified by the LMS due to other erroneous API calls being attempted prior to LMSInitialize and it is therefore necessary for the SCO to use this function to obtain error information under these circumstances.</td></tr>
<tr><td>2.1.2-12</td><td>The LMS shall provide an implementation for the API Adapter function LMSGetDiagnostic.</td></tr>
<tr><td>2.1.2-12.1</td><td>LMSGetDiagnostic shall accept a string parameter containing either a valid API Error Code, or an empty string (“”)</td></tr>
<tr><td>2.1.2-12.2</td><td>If LMSGetDiagnostic is called with a string parameter that contains a valid API Error Code, then the LMSGetDiagnostic function shall return a string value representing the LMS specific description corresponding to the API Error Code parameter provided.</td></tr>
<tr><td>2.1.2-12.3</td><td>If LMSGetDiagnostic is called with an empty string parameter (“”), then the LMSGetDiagnostic function shall return a string value representing the LMS specific description corresponding to the API Error Code that is currently set.</td></tr>
<tr><td>Note</td><td>The LMS must support the ability for a SCO to call LMSGetDiagnostic prior to calling LMSInitialize. This is necessary because error state (error code) information can be modified by the LMS due to other erroneous API calls being attempted prior to LMSInitialize and it is therefore necessary for the SCO to use this function to obtain error information under these circumstances.</td></tr>
</table>

Table 2.1.2a LMS Run-Time Environment API Conformance Requirements

2.1.3. LMS Run-Time Environment Data Model Conformance Requirements

The SCORM Version 1.1 Run-Time Environment Data Model is derived from the AICC CMI Data Model[6]. The data model is described in Section 3.4 of the SCORM Version 1.1.

In the SCORM Version 1.1, the Run-time Environment Data Model is tightly coupled to the API. The data model describes the information that may be exchanged between LMSs and SCOs via the API. Future versions of the SCORM may include additional or alternative communication mechanisms and/or data models. For this reason, the following data model conformance requirements are written such that they can stand-alone independently from the communication mechanism.

Conformance to the SCORM Version 1.1 Run-Time Environment API requirements is a prerequisite for conformance to the SCORM Version 1.1 Run-Time Environment Data Model requirements, since there is currently only one communication mechanism supported by the SCORM.

The following list provides a description of the key terms that are used in the requirements tables in this section to describe the LMS Run-time Environment Data Model implementation requirements:

- **Mandatory** – The LMS must implement the data model element. For example, in the requirement: "The LMS shall implement the cmi.core.student_id element (mandatory)", "mandatory" means that the LMS must implement the cmi.core.student_id element in order to be considered "LMS Run-Time Environment Conformant – Minimum" (LMS-RTE1).

- **Optional** – The LMS is not required to implement this element in order to be considered "LMS Run-Time Environment Conformant – Minimum" (LMS-RTE1).

- **Read-only** – The LMS must implement this element such that a SCO may only get (read) the value using the LMSGetValue API Adapter function. If the SCO attempts to set (write) a value for this element using the LMSSetValue API Adapter function, the LMS behaves according to the LMS Run-Time Environment API Conformance requirements and sets the appropriate API Error Code.

- **Write-only** – The LMS must implement this element such that a SCO may only set (write) a value for this element using the LMSSetValue API Adapter function. If the SCO attempts to get (read) the value for this element using the LMSGetValue API Adapter function, the LMS behaves according to the LMS Run-Time Environment API Conformance requirements and sets the appropriate API Error Code.

- **Read/write** – The LMS must implement this element such that a SCO may both set (write) and get (read) a value for this element using the LMSSetValue and LMSGetValue API Adapter functions, respectively.

- **Provide** – This signifies that the LMS must return a value that adheres to the stated requirement to the SCO. For example, the requirement: "Provide the value as a string of type CMIString255" means that when a SCO calls the LMSGetValue API Adapter function for the particular element, that the return value must conform to the CMI data type specified in the requirement.

- **Accept** – This signifies that the LMS must only accept data from the SCO that conforms to the stated requirement. For example, the requirement: "Accept only values for this element that are of type CMIVocabulary (Status)" signifies that the LMS must only accept values for the stated element that belong to the CMIVocabulary (Status) vocabulary. If the SCO attempts to set a value for the element that does not adhere to the CMIVocabulary (Status) vocabulary, then the LMS behaves according to the LMS Run-Time Environment API Conformance requirements and sets the appropriate API Error Code.

- **Initialize** – This signifies that the initial value provided to the SCO by the LMS should be determined based on the stated requirement.

This section contains two requirements tables. The first table, Table 2.1.3a, contains the requirements that must be met in order for the LMS to achieve certification to the 3 different LMS Run-Time Environment Conformance Categories:

- "SCORM Version 1.1 Run-Time Environment Conformant – Minimum" (LMS-RTE1)

- "SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Optional Data Model Elements" (LMS-RTE2)

- "SCORM Version 1.1 Run-Time Environment Conformant – Minimum with All Optional" (LMS-RTE3)

The second table, Table 2.1.3b, contains the SCORM Run-Time Environment Data Model Conformance requirements for conformant implementation of all of the data model elements. For example, requirement 2.1.3-3 states which data model elements must be implemented correctly by the LMS in order for the LMS to be certified "SCORM Version 1.1 Run-Time Environment Conformant – Minimum with All Optional" (LMS-RTE3). The requirements in Table 2.1.3b specify how those data model elements are to be implemented correctly.

Req No	Requirement
2.1.3-1	The LMS shall correctly implement support for all API Adapter functions as well as for **all** of the following mandatory data model elements in order to be considered "SCORM Version 1.1 Run-Time Environment Conformant – Minimum" (LMS-RTE1): • cmi.core._children • cmi.core.student_id • cmi.core.student_name • cmi.core.lesson_location • cmi.core.credit • cmi.core.lesson_status • cmi.core.entry • cmi.core.score._children • cmi.core.score.raw • cmi.core.total_time • cmi.core.exit • cmi.core.session_time • cmi.suspend_data • cmi.launch_data
2.1.3-2	The LMS shall correctly implement support for all API Adapter functions as well as for all of the mandatory data model elements as defined in requirement 2.1.3-1 and in addition, the LMS shall correctly implement support **one or more** of the following optional data model elements in order to be considered "SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Optional Data Model Elements" (LMS-RTE2): • cmi.core.score.max • cmi.core.score.min • cmi.core.lesson_mode • cmi.comments • cmi.comments_from_lms • cmi.objectives._children • cmi.objectives._count • cmi.objectives.n.id • cmi.objectives.n.score._children • cmi.objectives.n.score.raw • cmi.objectives.n.score.max • cmi.objectives.n.score.min • cmi.objectives.n.status

Req No	Requirement
	• cmi.student_data._children • cmi.student_data.mastery_score • cmi.student_data.max_time_allowed • cmi.student_data.time_limit_action • cmi.student_preference._children • cmi.student_preference.audio • cmi.student_preference.language • cmi.student_preference.speed • cmi.student_preference.text • cmi.interactions._children • cmi.interactions._count • cmi.interactions.n.id • cmi.interactions.n.objectives._count • cmi.interactions.n.objectives.n.id • cmi.interactions.n.time • cmi.interactions.n.type • cmi.interactions.n.correct_responses._count • cmi.interactions.n.correct_responses.n.pattern • cmi.interactions.n.weighting • cmi.interactions.n.student_response • cmi.interactions.n.result • cmi.interactions.n.latency
2.1.3-3	The LMS shall correctly implement support for all API Adapter functions as well as for all of the mandatory data model elements as defined in requirement 2.1.3-1 and in addition, the LMS shall implement support for **all** of the following optional data model elements in order to be considered "SCORM Version 1.1 Run-Time Environment Conformant – Minimum with All Optional" (LMS-RTE3): • cmi.core.score.max • cmi.core.score.min • cmi.core.lesson_mode • cmi.comments • cmi.comments_from_lms • cmi.objectives._children • cmi.objectives._count • cmi.objectives.n.id • cmi.objectives.n.score._children • cmi.objectives.n.score.raw • cmi.objectives.n.score.max

Req No	Requirement
	• cmi.objectives.n.score.min • cmi.objectives.n.status • cmi.student_data._children • cmi.student_data.mastery_score • cmi.student_data.max_time_allowed • cmi.student_data.time_limit_action • cmi.student_preference._children • cmi.student_preference.audio • cmi.student_preference.language • cmi.student_preference.speed • cmi.student_preference.text • cmi.interactions._children • cmi.interactions._count • cmi.interactions.n.id • cmi.interactions.n.objectives._count • cmi.interactions.n.objectives.n.id • cmi.interactions.n.time • cmi.interactions.n.type • cmi.interactions.n.correct_responses._count • cmi.interactions.n.correct_responses.n.pattern • cmi.interactions.n.weighting • cmi.interactions.n.student_response • cmi.interactions.n.result • cmi.interactions.n.latency

Table 2.1.3a LMS Run-Time Environment Data Model Conformance Requirements by Conformance Category

The following, Table 2.1.3b, contains the SCORM Run-Time Environment Data Model Conformance requirements for conformant implementation of all of the data model elements.

Req No	Requirement
2.1.3-4	The LMS shall implement the ***cmi.core*** category
2.1.3-4.1	The LMS shall implement the ***cmi.core._children*** element (**mandatory**). The LMS shall:

Req No	Requirement
2.1.3-4.1.1	Implement this element as a **read-only** element.
2.1.3-4.1.2	Provide the value as a string of type **CMIString255**.
2.1.3-4.1.3	Provide the value as a comma-separated list of all element names in this category that are supported by the LMS.
2.1.3-4.2	The LMS shall implement the ***cmi.core.student_id*** element (**mandatory**). The LMS shall:
2.1.3-4.2.1	Implement this element as a **read-only** element.
2.1.3-4.2.2	Provide the value as a string of type **CMIIdentifier**.
2.1.3-4.2.3	Initialize this element as the value that is used by the LMS to uniquely identify the student, based on the student registration information within the LMS.
2.1.3-4.2.4	Provide the value that is used by the LMS to uniquely identify the student.
2.1.3-4.3	The LMS shall implement the ***cmi.core.student_name*** element (**mandatory**). The LMS shall:
2.1.3-4.3.1	Implement this element as a **read-only** element.
2.1.3-4.3.2	Initialize this element using the name of the student, based on the student registration information within the LMS.
2.1.3-4.3.3	Provide the value as a string of type **CMIString255**.
2.1.3-4.3.4	Provide the value of the name of the student, based on the student registration information within the LMS.
2.1.3-4.4	The LMS shall implement the ***cmi.core.lesson_location*** element (**mandatory**). The LMS shall:
2.1.3-4.4.1	Implement this element as a **read/write** element.
2.1.3-4.4.2	Initialize this element to an empty string (“”).

Req No	Requirement
2.1.3-4.4.3	Provide the value as a string of type **CMIString255**.
2.1.3-4.4.4	Accept values for this element that are of type **CMIString255**.
2.1.3-4.5	The LMS shall implement the ***cmi.core.credit*** element (**mandatory**). The LMS shall:
2.1.3-4.5.1	Implement this element as a **read-only** element.
2.1.3-4.5.2	Initialize the value of this element to: • "credit" if the student is taking the SCO for credit. • "no-credit" if the student is not taking the SCO for credit.
2.1.3-4.5.3	Provide the value as a string of type **CMIVocabulary (Credit)**.
2.1.3-4.5.4	Determine whether or not the student is taking the course for credit or no-credit based on the student registration process and/or data managed by the LMS.
2.1.3-4.6	The LMS shall implement the ***cmi.core.lesson_status*** element (**mandatory**). The LMS shall:
2.1.3-4.6.1	Implement this element as a **read/write** element.
2.1.3-4.6.2	Initialize this element to "not attempted".
2.1.3-4.6.3	Provide the value as a **CMIVocabulary (Status)**.
2.1.3-4.6.4	Accept only values for this element that are of type **CMIVocabulary (Status)**.
2.1.3-4.6.5	Not accept **CMIVocabulary (Status)** value of "not attempted" for this element from a SCO. (This value can only be set by the LMS).
2.1.3-4.6.6	***(Not tested)*** ***The following requirement is deferred until SCORM Version 1.2 Certification Testing. This requirement depends on the availability of mastery_score for each SCO from the CSF. Since LMSs are not required to implement the ability to import courses using a CSF XML document, this requirement can not be tested. It is anticipated that SCORM 1.2 will include a mastery score element for SCOs in the***

Req No	Requirement
	content packaging manifest. If the value for this element is not set to "incomplete" by the SCO, then the LMS shall re-evaluate and change the value based on the following: • If there is a mastery score in the CSF, the LMS can change the status to either passed or failed depending on the student's score (cmi.core.score.raw) compared to the mastery score. • If there is no mastery score in the CSF, the LMS cannot override SCO determined status. • If the student is taking the SCO for no-credit, (cmi.core.credit = "credit") there is no change to the lesson_status, with one exception. If the lesson_mode (cmi.core.lesson_mode) is "browse", the lesson_status may change to "browsed" even if the cmi.core.credit is set to no-credit.
2.1.3-4.7	The LMS shall implement the ***cmi.core.entry*** element (**mandatory**). The LMS shall:
2.1.3-4.7.1	Implement this element as a **read-only** element.
2.1.3-4.7.2	Initialize the value of this element to: • "ab-initio" upon the first launch of the SCO. • "resume" upon subsequent launches of the SCO • "" (empty string) if the value of the cmi.core.lesson_mode element is "review" and the value of the cmi.core.lesson_status element is either "passed", "failed", or "completed".
2.1.3-4.7.3	Provide the value as a string of type **CMIVocabulary (Entry)**.
2.1.3-4.8	The LMS shall implement the ***cmi.core.score*** category (**mandatory**).
2.1.3-4.8.1	The LMS shall implement the ***cmi.core.score._children*** element (**mandatory**). The LMS shall:
2.1.3-4.8.1.1.	Implement this element as a **read-only** element.
2.1.3-4.8.1.2.	Provide the value as a string of type **CMIString255**.

Req No	Requirement
2.1.3-4.8.1.3.	Provide the value as a comma-separated list of all element names in this category that are supported by the LMS.
2.1.3-4.8.2	The LMS shall implement the ***cmi.core.score.raw*** element (**mandatory**). The LMS shall:
2.1.3-4.8.2.1.	Implement this element as a **read/write** element.
2.1.3-4.8.2.2.	Initialize this element to an empty string (“”).
2.1.3-4.8.2.3.	Provide the value as a string of type **CMIDecimal or CMIBlank** (if no value is set).
2.1.3-4.8.2.4.	Accept values for this element that are of type **CMIDecimal** or **CMIBlank**.
2.1.3-4.8.3	The LMS shall implement the ***cmi.core.score.max*** element **(optional)**. The LMS shall:
2.1.3-4.8.3.1.	Implement this element as a **read/write** element.
2.1.3-4.8.3.2.	Initialize this element to an empty string (“”).
2.1.3-4.8.3.3.	Provide the value as a string of type **CMIDecimal or CMIBlank** (if no value is set).
2.1.3-4.8.3.4.	Accept values for this element that are of type **CMIDecimal** or **CMIBlank**.
2.1.3-4.8.4	The LMS shall implement the ***cmi.core.score.min*** element **(optional)**. The LMS shall:
2.1.3-4.8.4.1.	Implement this element as a **read/write** element.
2.1.3-4.8.4.2.	Initialize this element to an empty string (“”).
2.1.3-4.8.4.3.	Provide the value as a string of type **CMIDecimal or CMIBlank** (if no value is set).
2.1.3-4.8.4.4.	Accept values for this element that are of type **CMIDecimal** or **CMIBlank**.

Req No	Requirement
2.1.3-4.9	The LMS shall implement the ***cmi.core.total_time*** element (**mandatory**). The LMS shall:
2.1.3-4.9.1	Implement this element as a **read-only** element.
2.1.3-4.9.2	Initialize this element to a string containing ("0000:00:00.00")
2.1.3-4.9.3	Provide the value as a string of type **CMITimespan**.
2.1.3-4.9.4	Provide this value as the accumulation of time spent by the student in the SCO by adding the value of the cmi.core.session_time element set by the SCO during each access of the SCO to the value of this element, if provided by the SCO. If multiple cmi.core.session_time values are received by the LMS during one SCO access, the LMS should only use the last set for accumulation. The LMS should not accumulate the cmi.core.session_time values set by the SCO during the one access. The LMS shall only accumulate cmi.core.total_time on an LMSFinish() call.
2.1.3-4.10	The LMS shall implement the ***cmi.core.lesson_mode*** element **(optional)**. The LMS shall:
2.1.3-4.10.1	Implement this element as a **read-only** element
2.1.3-4.10.2	Initialize the value of this element to one of the following values, based on the capabilities of the LMS: • "browse" • "normal" • "review"
2.1.3-4.10.3	Provide the value as a string of type **CMIVocabulary (Mode)**.
2.1.3-4.11	The LMS shall implement the ***cmi.core.exit*** element (**mandatory**). The LMS shall:
2.1.3-4.11.1	Implement this element as a **write-only** element.
2.1.3-4.11.2	Accept the value only as a string of type **CMIVocabulary (Exit)**.

Req No	Requirement
2.1.3-4.12	The LMS shall implement the ***cmi.core.session_time*** element (**mandatory**). The LMS shall:
2.1.3-4.12.1	Implement this element as a **write-only** element.
2.1.3-4.12.2	Accept the value only as a string of type **CMITimespan**.
2.1.3-4.12.3	Accumulate the last provided value of this element for a given SCO by adding the last provided value to the cmi.core.total_time element upon the receipt of an LMSFinish() call.
2.1.3-5	The LMS shall implement the ***cmi.suspend_data*** element (**mandatory**). The LMS shall:
2.1.3-5.1	Implement this element as a **read/write** element.
2.1.3-5.2	Initialize this element to an empty string (“”).
2.1.3-5.3	Provide the value as a string of type **CMIString4096**.
2.1.3-5.4	Accept values for this element that are of type **CMIString4096**.
2.1.3-6	The LMS shall implement the ***cmi.launch_data*** element (**mandatory**). The LMS shall:
2.1.3-6.1	Implement this element as a **read-only** element.
2.1.3-6.2	Initialize the value of this element as follows: • If the LMS is able to import content using a CSF XML document, and the CSF element content|block|sco|launch|dataFromLMS is present, for the SCO, then this value is used for initialization. If the content|block|sco|launch|dataFromLMS CSF element is not present, then the value of this element is initialized to an empty string (“”). • If the LMS is not able to import content using a CSF XML document, then this element is initialized to an empty string (“”).
2.1.3-6.3	Provide the value as a string of type **CMIString4096**.

Req No	Requirement
2.1.3-7	The LMS shall implement the ***cmi.comments*** element **(optional)**. The LMS shall:
2.1.3-7.1	Implement this element as a **read/write** element.
2.1.3-7.2	Initialize this element to an empty string (“”) prior to the first entry into the SCO. (Note: first entry is defined as the first entry by a single student, including all sessions.)
2.1.3-7.3	Provide the value as a string of type **CMIString4096**.
2.1.3-7.4	Accept values for this element that are of type **CMIString4096**.
2.1.3-7.5	Append each valid value set by the SCO to the values previously set by the SCO. (This data model element accumulates the comments set by the SCO, as opposed to overwriting previously set values).
2.1.3-8	The LMS shall implement the ***cmi.comments_from_lms*** element **(optional)**. The LMS shall:
2.1.3-8.1	Implement this element as a **read-only** element.
2.1.3-8.2	Initialize this element using the value entered into the LMS (if supported by the LMS).
2.1.3-8.3	Provide the value as a string of type **CMIString4096**.
2.1.3-9	The LMS shall implement the ***cmi.objectives*** category **(optional)**.
2.1.3-9.1	The LMS shall implement the ***cmi.objectives._children*** element (**mandatory** if any of the defined elements of the category are supported by the LMS). The LMS shall:
2.1.3-9.1.1	Implement this element as a **read-only** element.
2.1.3-9.1.2	Provide the value as a string of type **CMIString255**.
2.1.3-9.1.3	Provide the value as a comma-separated list of all element names in this category that are supported by the LMS.

Req No	Requirement
2.1.3-9.2	The LMS shall implement the ***cmi.objectives._count*** element (**mandatory** if any of the defined elements of the category are supported by the LMS). The LMS shall:
2.1.3-9.2.1	Implement this element as a **read-only** element.
2.1.3-9.2.2	Provide the value as a string of type **CMIInteger**.
2.1.3-9.2.3	Provide the number of entries in the cmi.objectives list that is maintained by the LMS for the corresponding SCO.
2.1.3-9.3	The LMS shall implement the ***cmi.objectives.n.id*** element **(optional)**. The LMS shall:
2.1.3-9.3.1	Implement this element as a **read/write** element.
2.1.3-9.3.2	Initialize this element to an empty string ("") prior to the first entry into the SCO.
2.1.3-9.3.3	Provide the value as a string of type **CMIIdentifier**
2.1.3-9.3.4	Accept values for this element that are of type **CMIIdentifier**
2.1.3-9.3.5	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list
2.1.3-9.4	The LMS shall implement the ***cmi.objectives.n.score*** category **(optional)**.
2.1.3-9.4.1	The LMS shall implement the ***cmi.objectives.n.score._children*** element (**mandatory** if any of the defined elements of the category are supported by the LMS). The LMS shall:
2.1.3-9.4.1.1.	Implement this element as a **read-only** element.
2.1.3-9.4.1.2.	Provide the value as a string of type **CMIString255**.
2.1.3-9.4.1.3.	Provide the value as a comma-separated list of all element names in this category that are supported by the LMS.
2.1.3-9.4.1.4.	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list

Req No	Requirement
2.1.3-9.4.2	The LMS shall implement the ***cmi.objectives.n.score.raw*** element **(optional)**. The LMS shall:
2.1.3-9.4.2.1.	Implement this element as a **read/write** element.
2.1.3-9.4.2.2.	Initialize this element to an empty string (“”).
2.1.3-9.4.2.3.	Provide the value as a string of type **CMIDecimal or CMIBlank** (if no value is set).
2.1.3-9.4.2.4.	Accept values for this element that are of type **CMIDecimal** or **CMIBlank**.
2.1.3-9.4.2.5.	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list.
2.1.3-9.4.3	The LMS shall implement the ***cmi.objectives.n.score.max*** element **(optional)**. The LMS shall:
2.1.3-9.4.3.1.	Implement this element as a **read/write** element.
2.1.3-9.4.3.2.	Initialize this element to an empty string (“”).
2.1.3-9.4.3.3.	Provide the value as a string of type **CMIDecimal or CMIBlank** (if no value is set).
2.1.3-9.4.3.4.	Accept values for this element that are of type **CMIDecimal** or **CMIBlank**.
2.1.3-9.4.3.5.	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list.
2.1.3-9.4.4	The LMS shall implement the ***cmi.objectives.n.score.min*** element **(optional)**. The LMS shall:
2.1.3-9.4.4.1.	Implement this element as a **read/write** element.
2.1.3-9.4.4.2.	Initialize this element to an empty string (“”).
2.1.3-9.4.4.3.	Provide the value as a string of type **CMIDecimal or CMIBlank** (if no value is set).

Req No	Requirement
2.1.3-9.4.4.4.	Accept values for this element that are of type **CMIDecimal** or **CMIBlank**.
2.1.3-9.4.4.5.	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list.
2.1.3-9.5	The LMS shall implement the ***cmi.objectives.n.status*** element **(optional)**. The LMS shall:
2.1.3-9.5.1	Implement this element as a **read/write** element.
2.1.3-9.5.2	Initialize this element to "not attempted".
2.1.3-9.5.3	Provide the value as a **CMIVocabulary (Status)**.
2.1.3-9.5.4	Accept only values for this element that are of type **CMIVocabulary (Status)**.
2.1.3-9.5.5	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list.
2.1.3-10	The LMS shall implement the ***cmi.student_data*** category **(optional)**.
2.1.3-10.1	The LMS shall implement the ***cmi.student_data._children*** element (**mandatory** if any of the defined elements of the category are supported by the LMS). The LMS shall:
2.1.3-10.1.1	Implement this element as a **read-only** element.
2.1.3-10.1.2	Provide the value as a string of type **CMIString255**.
2.1.3-10.1.3	Provide the value as a comma-separated list of all element names in this category that are supported by the LMS.
2.1.3-10.2	The LMS shall implement the ***cmi.student_data.mastery_score*** element **(optional)**. The LMS shall:
2.1.3-10.2.1	Implement this element as a **read-only** element.

Req No	Requirement
2.1.3-10.2.2	Initialize the value of this element as follows: • If the LMS is able to import content using a CSF XML document, and the CSF element content|block|sco|masteryScore is present, for the SCO, then this value is used for initialization. If the content|block|sco|masteryScore CSF element is not present, then the value of this element is initialized to an empty string (“”). • If the LMS is not able to import content using a CSF XML document, then this element is initialized to an empty string (“”).
2.1.3-10.2.3	Provide the value as a string of type **CMIDecimal** or **CMIBlank**.
2.1.3-10.3	The LMS shall implement the ***cmi.student_data.max_time_allowed*** element **(optional)**. The LMS shall:
2.1.3-10.3.1	Implement this element as a **read-only** element.
2.1.3-10.3.2	Initialize the value of this element as follows: • If the LMS is able to import content using a CSF XML document, and the CSF element content|block|sco|timeLimit|maxTimeAllowed is present, for the SCO, then this value is used for initialization. If the content|block|sco|timeLimit|maxTimeAllowed CSF element is not present, then the value of this element is initialized to an empty string (“”). • If the LMS is not able to import content using a CSF XML document, then this element is initialized to an empty string (“”).
2.1.3-10.3.3	Provide the value as a string of type **CMITimespan** or **CMIBlank**.
2.1.3-10.4	The LMS shall implement the ***cmi.student_data.time_limit_action*** element **(optional)**. The LMS shall:
2.1.3-10.4.1	Implement this element as a **read-only** element.

Req No	Requirement
2.1.3-10.4.2	Initialize the value of this element as follows: • If the LMS is able to import content using a CSF XML document, and the CSF element content|block|sco|timeLimit|timeLimitAction is present, for the SCO, then this value is used for initialization. If the content|block|sco|timeLimit|timeLimitAction CSF element is not present, then the value of this element is initialized to an empty string (""). • If the LMS is not able to import content using a CSF XML document, then this element is initialized to an empty string ("").
2.1.3-10.4.3	Provide the value as a string of type **CMIVocabulary (Time Limit Action)** or **CMIBlank**.
2.1.3-11	The LMS shall implement the ***cmi.student_preference*** category **(optional)**.
2.1.3-11.1	The LMS shall implement the ***cmi.student_preference._children*** element (**mandatory** if any of the defined elements of the category are supported by the LMS). The LMS shall:
2.1.3-11.1.1	Implement this element as a **read-only** element.
2.1.3-11.1.2	Provide the value as a string of type **CMIString255**.
2.1.3-11.1.3	Provide the value as a comma-separated list of all element names in this category that are supported by the LMS.
2.1.3-11.2	The LMS shall implement the ***cmi.student_preference.audio*** element **(optional)**. The LMS shall:
2.1.3-11.2.1	Implement this element as a **read/write** element.
2.1.3-11.2.2	Initialize this element to "0" (zero) prior to the first entry into the SCO.
2.1.3-11.2.3	Provide the value as a string of type **CMISInteger**.
2.1.3-11.2.4	Accept values for this element that are of type **CMISInteger** between the range of –1 to 100.

Req No	Requirement
2.1.3-11.3	The LMS shall implement the ***cmi.student_preference.language*** element **(optional)**. The LMS shall:
2.1.3-11.3.1	Implement this element as a **read/write** element.
2.1.3-11.3.2	Initialize this element to an empty string (“”) prior to the first entry into the SCO.
2.1.3-11.3.3	Provide the value as a string of type **CMIString255**.
2.1.3-11.3.4	Accept values for this element that are of type **CMIString255**.
2.1.3-11.4	The LMS shall implement the ***cmi.student_preference.speed*** element **(optional)**. The LMS shall:
2.1.3-11.4.1	Implement this element as a **read/write** element.
2.1.3-11.4.2	Initialize this element to “0” (zero) prior to the first entry into the SCO.
2.1.3-11.4.3	Provide the value as a string of type **CMISInteger**.
2.1.3-11.4.4	Accept values for this element that are of type **CMISInteger** between the range of –1 to 100.
2.1.3-11.5	The LMS shall implement the ***cmi.student_preference.text*** element **(optional)**. The LMS shall:
2.1.3-11.5.1	Implement this element as a **read/write** element.
2.1.3-11.5.2	Initialize this element to “0” (zero) prior to the first entry into the SCO.
2.1.3-11.5.3	Provide the value as a string of type **CMISInteger**.
2.1.3-11.5.4	Accept values for this element that are of type **CMISInteger** of –1, 0, or 1.
2.1.3-12	The LMS shall implement the ***cmi.interactions*** category **(optional)**.

Req No	Requirement
2.1.3-12.1	The LMS shall implement the ***cmi.interactions._children*** element (**mandatory** if any of the defined elements of the category are supported by the LMS). The LMS shall:
2.1.3-12.1.1	Implement this element as a **read-only** element.
2.1.3-12.1.2	Provide the value as a string of type **CMIString255**.
2.1.3-12.1.3	Provide the value as a comma-separated list of all element names in this category that are supported by the LMS.
2.1.3-12.2	The LMS shall implement the ***cmi.interactions._count*** element (**mandatory** if any of the defined elements of the category are supported by the LMS). The LMS shall:
2.1.3-12.2.1	Implement this element as a **read-only** element.
2.1.3-12.2.2	Provide the value as a string of type **CMIInteger**.
2.1.3-12.2.3	Provide the number of entries in the cmi.interactions list that is maintained by the LMS for the corresponding SCO.
2.1.3-12.3	The LMS shall implement the ***cmi.interactions.n.id*** element **(optional)**. The LMS shall:
2.1.3-12.3.1	Implement this element as a **write-only** element.
2.1.3-12.3.2	Provide the value as a string of type **CMIIdentifier**.
2.1.3-12.3.3	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list.
2.1.3-12.4	The LMS shall implement the ***cmi.interactions.n.objectives*** element **(optional)**. The LMS shall:
2.1.3-12.4.1	The LMS shall implement the ***cmi.interactions.n.objectives._count*** element **(optional)**. The LMS shall:

Req No	Requirement
2.1.3-12.4.1.1	Implement this element as a **read-only** element.
2.1.3-12.4.1.2	Provide the value as a string of type **CMIInteger**.
2.1.3-12.4.1.3	Provide the number of entries in the cmi.interactions.n.objectives list that is maintained by the LMS for the corresponding SCO.
2.1.3-12.4.2	The LMS shall implement the ***cmi.interactions.n.objectives.n.id*** element **(optional)**. The LMS shall:
2.1.3-12.4.2.1	Implement this element as a **write-only** element.
2.1.3-12.4.2.2	Accept values for this element that are of type **CMIIdentifier.**
2.1.3-12.4.2.3	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list
2.1.3-12.5	The LMS shall implement the ***cmi.interactions.n.time*** element **(optional)**. The LMS shall:
2.1.3-12.5.1	Implement this element as a **write-only** element.
2.1.3-12.5.2	Accept values for this element that are of type **CMITime.**
2.1.3-12.5.3	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list.
2.1.3-12.6	The LMS shall implement the ***cmi.interactions.n.type*** element **(optional)**. The LMS shall:
2.1.3-12.6.1	Implement this element as a **write-only** element.
2.1.3-12.6.2	Accept values for this element that are of type **CMIVocabulary (Interaction)**
2.1.3-12.6.3	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list.
2.1.3-12.7	The LMS shall implement the ***cmi.interactions.n.correct_responses*** element **(optional)**. The LMS shall:

Req No	Requirement
2.1.3-12.7.1	The LMS shall implement the ***cmi.interactions.n.correct_responses._count*** element (**mandatory** if any of the defined elements of the category are supported by the LMS). The LMS shall:
2.1.3-12.7.1.1	Implement this element as a **read-only** element.
2.1.3-12.7.1.2	Provide the value as a string of type **CMIInteger**.
2.1.3-12.7.1.3	Provide the number of entries in the cmi.interactions.n.correct_responses list that is maintained by the LMS for the corresponding SCO.
2.1.3-12.7.1.4	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list.
2.1.3-12.7.2	The LMS shall implement the ***cmi.interactions.n.correct_responses.n.pattern*** element **(optional)**. The LMS shall:
2.1.3-12.7.2.1	Implement this element as a **write-only** element.
2.1.3-12.7.2.2	Provide the value as a string of type **CMIFeedback**.
2.1.3-12.7.2.3	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list.
2.1.3-12.8	The LMS shall implement the ***cmi.interactions.n.weighting*** element **(optional)**. The LMS shall:
2.1.3-12.8.1	Implement this element as a **write-only** element.
2.1.3-12.8.2	Accept values for this element that are of type **CMIDecimal.**
2.1.3-12.8.3	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list.
2.1.3-12.9	The LMS shall implement the ***cmi.interactions.n.student_response*** element **(optional)**. The LMS shall:
2.1.3-12.9.1	Implement this element as a **write-only** element.

Req No	Requirement
2.1.3-12.9.2	Accept values for this element that are of type **CMIFeedback.**
2.1.3-12.9.3	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list.
2.1.3-12.10	The LMS shall implement the ***cmi.interactions.n.result*** element **(optional)**. The LMS shall:
2.1.3-12.10.1	Implement this element as a **write-only** element.
2.1.3-12.10.2	Accept values for this element that are of type **CMIVocabulary (Result).**
2.1.3-12.10.3	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list.
2.1.3-12.11	The LMS shall implement the ***cmi.interactions.n.latency*** element **(optional)**. The LMS shall:
2.1.3-12.11.1	Implement this element as a **write-only** element.
2.1.3-12.11.2	Accept values for this element that are of type **CMITimespan.**
2.1.3-12.11.3	Accept only elements where the list index (n) is less than or equal to 1 greater than the number of elements currently in the list.
2.1.3-13	The LMS shall implement the defined SCORM Run-time Environment Data Model data types and restricted vocabularies as follows:
2.1.3-13.1	CMIBlank: Value shall hold an empty string (“”).
2.1.3-13.2	CMIBoolean: Value shall contain a restricted vocabulary of two words – “true” and “false”.
2.1.3-13.3	CMIDecimal: Value shall contain a decimal number.
2.1.3-13.4	CMIFeedback: Value shall contain data corresponding to the specific CMIVocabulary Interaction type defined for the interaction (i.e. cmi.interactions.n.type) as follows:

<table>
<tr><th>Req No</th><th>Requirement</th></tr>
<tr><td></td><td>
<table>
<tr><th>Type</th><th>Description</th></tr>
<tr><td>true-false</td><td>Feedback is one of the following single characters: “0”,”1”,”t”, or “f”.</td></tr>
<tr><td>choice</td><td>Feedback is one or more single characters separated by a comma. Legal characters are “0” to “9” and “a” to “z”. If all the characters must be chosen to assume the feedback is correct, then the comma-separated list must be surrounded by curly brackets: { }</td></tr>
<tr><td>fill-in</td><td>Any alpha-numeric string up to 255 characters in length. After the first letter spaces are significant.</td></tr>
<tr><td>numeric</td><td>CMIDecimal</td></tr>
<tr><td>likert</td><td>Single character. Legal characters are 0 to 9 and a to z.</td></tr>
<tr><td>matching</td><td>One or more pairs of identifiers. Each identifier is a single letter or number (0 to 9 and a to z). The identifiers in a pair are separated by a period. Commas separate the pairs. If all pairs must be matched correctly to consider the interaction correct, then the comma separated list of pairs are surrounded by curly brackets: { }.</td></tr>
<tr><td>performance</td><td>This is a very flexible format. Essentially an alphanumeric string of 255 characters or less.</td></tr>
<tr><td>sequencing</td><td>A series of single characters separated by commas. Legal characters are 0 to 9 and a to z. The order of the characters determines the correctness of the feedback.</td></tr>
</table>
</td></tr>
<tr><td>2.1.3-13.5</td><td>CMIIdentifier:
Value shall contain an alphanumeric group of characters
Value shall contain no white space
Value shall contain no unprintable characters
Value’s length shall not exceed 255 characters</td></tr>
<tr><td>2.1.3-13.6</td><td>CMISInteger: Value shall contain a signed integer number between –32768 and 32768</td></tr>
<tr><td>2.1.3-13.7</td><td>CMIString255: Value shall contain a set of ASCII characters
Value’s maximum length shall not exceed 255 characters</td></tr>
</table>

Req No	Requirement
2.1.3-13.8	CMIString4096: Value shall contain a set of ASCII characters Value's maximum length shall not exceed 4096 characters
2.1.3-13.9	CMITime: Value is a chronological point in a 24 hour clock • Value shall be represented in the format HH:MM:SS.SS where HH signifies hours, MM signifies minutes and SS.SS signifies seconds. • Value's hours shall contain exactly 2 digits between 00 – 23 • Value's minutes shall contain exactly 2 digits between 00 – 59 • Value's seconds shall contain at least 2 digits between 00 – 59 and may additionally (optionally) contain a decimal point and 1 or 2 additional digits after the decimal

2.1.3-13.10	CMITimespan: Value is a length of time
	• Value shall be represented in the format HHHH:MM:SS.SS where HHHH signifies hours, MM signifies minutes and SS.SS signifies seconds.
	• Value's hours shall contain:
	  ○ A minimum of 2 digits
	  ○ A maximum of 4 digits
	  ○ A value between 0000 – 9999
	• Value's minutes shall contain:
	  ○ Exactly 2 digits
	  ○ A value between 00 – 59
	• Value's seconds shall contain:
	  ○ A minimum of 2 digits
	  ○ Value may (optionally) contain a decimal point and 1 or 2 additional digits after the decimal
	  ○ Value between 00 - 59

2.1.3-13.11	CMIVocabulary: Value shall contain a list of restricted vocabulary word/phrases, depending on the defined vocabulary type as follows:
2.1.3-13.11.1	CMIVocabulary **Mode** shall contain the following values: • "normal" • "review" • "browse"
2.1.3-13.11.2	CMIVocabulary **Status** shall contain the following values: • "passed" • "completed" • "failed" • "incomplete" • "browsed" • "not attempted"

Req No	Requirement
2.1.3-13.11.3	CMIVocabulary **Exit** shall contain the following values: • "time-out" • "suspend" • "logout" • "" (empty string)
2.1.3-13.11.4	CMIVocabulary **Credit** shall contain the following values: • "credit" • "no-credit"
2.1.3-13.11.5	CMIVocabulary **Entry** shall contain the following values: • "ab-initio" • "resume" • "" (empty string)
2.1.3-13.11.6	CMIVocabulary **Interaction** shall contain the following values: • "true-false" • "choice" • "fill-in" • "matching" • "performance" • "likert" • "sequencing" • "numeric"
2.1.3-13.11.7	CMIVocabulary **Result** shall contain the following values: • "correct" • "wrong" • "unanticipated" • "neutral" • X.X – (CMIDecimal)
2.1.3-13.11.8	CMIVocabulary **Time Limit Action** shall contain the following values: • "exit,message" • "exit,no message" • "continue,message" • "continue,no message"
2.1.3-14	The LMS shall maintain separate instances of the data model element values for each SCO.

Table 2.1.3b LMS Run-Time Environment Data Model Conformance Requirements

2.2. SCO Run-Time Environment Conformance Requirements

In order to become certified as SCORM Run-Time Environment Conformant, a SCO is required to support the Run-Time Environment that is described in Section 3 of the SCORM Version 1.1. There are three main components of the SCORM Run-time Environment:

4. Launch
5. Application Program Interface (API)
6. Data Model

The conformance requirements for SCO Run-Time Environment Conformance are broken down into the following sections to address each of the Run-time environment components. They are as follows:

- Section 2.2.1 describes the Launch and API related conformance requirements
- Section 2.2.2 describes the Data Model related conformance requirements

As described in the Conformance Matrix in Section 1, the SCO Run-Time Environment Conformance requirements are defined in terms of four categories. The four categories are mutually exclusive. A SCO, if conformant, will be designated as conformant within one and only one of these categories. The categories are:

- SCORM Version 1.1 Run-Time Environment Conformant – Minimum (SCO-RTE1)
- SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Mandatory Data Model Elements(SCO-RTE1+Mandatory)
- SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Optional Data Model Elements(SCO-RTE1+Optional)
- SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Mandatory and Some Optional Data Model Elements(SCO-RTE1+Mandatory+Optional)

2.2.1. SCO Run-Time Environment Launch and API Conformance Requirements

Once launched by an LMS, the SCO is responsible for finding the API Adapter and correctly issuing the API function calls. The specific requirements that will be verified by the certification testing process are as follows:

Req No	Requirement
2.2.1-1	When launched by a known conformant LMS, the SCO shall find the LMS API Adapter DOM object named “API” by searching the parent and opener DOM window hierarchy.
2.2.1-2	The SCO shall be implemented such that it does not require that it be the top-level window in the DOM window hierarchy upon launch. The SCO must not contain DOM documents that reference relative documents within the SCO using the window.top DOM object.
2.2.1-3	The SCO shall call the LMSInitialize API Adapter function upon launch of the SCO and before calling any other API Adapter function, except for LMSGetLastError, LMSGetErrorString and/or LMSGetDiagnostic.
2.2.1-3.1	The SCO shall call the LMSInitialize API Adapter function with a single string parameter that contains an empty string value (i.e. “”).
2.2.1-3.2	***(Not tested)*** The SCO should accept a CMIBoolean string return value when calling the LMSInitialize API Adapter function.
2.2.1-4	The SCO shall call LMSInitialize one and only one time per launch of the SCO.
2.2.1-5	The SCO shall call the LMSFinish API Adapter function when it is finished communicating with the LMS.
2.2.1-5.1	The SCO shall call the LMSFinish API Adapter function with a single string parameter that contains an empty string value (i.e. “”).
2.2.1-5.2	***(Not tested)*** The SCO should accept a CMIBoolean string return value when calling the LMSFinish API Adapter function.
2.2.1-6	If the SCO implements the ability to set data to the LMS that launched the SCO, then the SCO shall use the LMS provided API Adapter function LMSSetValue to set (write) values for SCORM Run-Time Environment data model elements to the LMS (as opposed to HACP or other communication mechanisms).
2.2.1-6.1	The SCO shall call the LMSSetValue API Adapter function with two string parameters.
2.2.1-6.1.1	The first string parameter is the fully qualified, case sensitive string containing the name of the data model element requested to be set by the SCO.

Req No	Requirement
2.2.1-6.1.2	The second parameter is the string containing the value of the data model element that the SCO is requesting to be set.
2.2.1-6.2	***(Not tested)*** The SCO should accept a CMIBoolean string return value when calling the LMSSetValue API Adapter function.
2.2.1-7	If the SCO implements the ability to get data from the LMS that launched the SCO, then the SCO shall use the LMS provided API Adapter function LMSGetValue to get (read) values for SCORM Run-Time Environment data model elements from the LMS (as opposed to HACP or other communication mechanisms).
2.2.1-7.1	The SCO shall call the LMSGetValue API Adapter function with one string parameters that contains the fully qualified, case sensitive name of the data model element requested to be gotten (read) by the SCO.
2.2.1-7.2	***(Not tested)*** The SCO should accept a string return value when calling the LMSGetValue API Adapter function.
2.2.1-8	If the SCO has a need to determine error codes set by LMS provided API Adapter functions that are called by the SCO, then, the SCO shall use the LMS provided API Adapter function LMSGetLastError.
2.2.1-8.1	The SCO shall call the LMSGetLastError API Adapter function with no parameters.
2.2.1-8.2	***(Not tested)*** The SCO should accept a string return value that contains a valid API Error Code string when calling the LMSGetLastError API Adapter function.
2.2.1-9	If the SCO has a need to determine error code description that corresponds to valid API Error Code then, the SCO shall use the LMS provided API Adapter function LMSGetErrorString.
2.2.1-9.1	The SCO shall call the LMSGetErrorString API Adapter function with one string parameter that contains a valid API Error Code.
2.2.1-9.2	***(Not tested)*** The SCO should accept a string return value that contains a valid API Error Code Description string when calling the LMSGetErrorString API Adapter function.

Req No	Requirement
2.2.1-10	If the SCO has a need to determine LMS specific diagnostic information related to a valid API Error Code then, the SCO shall use the LMS provided API Adapter function LMSGetDiagnostic.
2.2.1-10.1	The SCO shall call the LMSGetDiagnostic API Adapter function with one string parameter that contains either a valid API Error Code or an empty string (“”). (Note: If an empty string is passed to the function, then the diagnostic text associated to the currently set API Error Code is returned.)
2.2.1-10.2	***(Not tested)*** The SCO should accept a string return value that contains an LMS specific description string when calling the LMSGetDiagnostic API Adapter function.
2.2.1-11	If the SCO has a need to assure that data model element values set by the SCO are persisted, without calling LMSFinish, then the SCO shall call LMSCommit API Adapter function. (Note: The LMS may persist data model element state automatically, but this is a way for the SCO to be assured that data model element state is persisted without having to exit the SCO/call LMSFinish.)
2.2.1-11.1	The SCO shall call the LMSCommit API Adapter function with one string parameter that contains an empty string (“”).
2.2.1-11.2	***(Not tested)*** The SCO should accept a CMIBoolean string return value when calling the LMSCommit API Adapter function.

Table 2.2.1a SCO Run-Time Environment API Conformance Requirements

2.2.2. SCO Run-Time Environment Data Model Conformance Requirements

LMSs are required to implement all mandatory SCORM Run-Time Environment data model elements in order to be considered SCORM Run-Time Environment Conformant. It should also be noted that LMSs are NOT required to implement the optional data model elements. SCOs that exchange data with LMSs should take this into account and downgrade gracefully if a requested data model element is not implemented by the LMS. This is not a requirement for conformance on the part of the SCO and will not be tested.

A SCO is not required to exchange data with an LMS in order to achieve “SCORM Version 1.1 Run-Time Environment Conformant – Minimum” status. Additionally, a SCO may attempt to exchange data with LMSs using elements that are not part of the CMI data model. Doing so is not recommended because interoperability is hindered, however this will not prevent the SCO from achieving “SCORM Version 1.1 Run-Time Environment Conformant – Minimum” status.

If the SCO does implement the ability to exchange data with an LMS, it must use the defined SCORM Run-Time Environment data model elements, within the bounds of the requirements as defined in the following table (Table 2.2.2a) in order to achieve "SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Mandatory Data Model Elements" and/or "SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Optional Data Model Elements" status.

The mandatory and/or optional data model elements that are correctly implemented, if any, will determine the SCO conformance status. If the SCO, through it's correct usage of the Run-Time Environment API has achieved "SCORM Version 1.1 Run-Time Environment Conformant – Minimum"(SCO-RTE1) status, then, as defined in requirements 2.2.2-1 and 2.2.2-2, the SCO may also achieve SCO-RTE+Mandatory and/or SCO-RTE+Optional conformance status.

Please note that the SCO is determined to be SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Optional and Some Mandatory Elements" (SCO-RTE1+Mandatory+Optional) if it satisfies the requirements for both SCO-RTE1+Mandatory and SCO-RTE1+Optional.

Req No	Requirement
2.2.2-1	If the SCO, through it's correct usage of the Run-Time Environment API, has achieved "SCORM Version 1.1 Run-Time Environment Conformant – Minimum"(SCO-RTE1) status, then in order to also be considered "SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Mandatory" (SCO–RTE1+Mandatory), the SCO shall correctly set or get one or more of the following LMS Mandatory data model elements: • cmi.core._children • cmi.core.student_id • cmi.core.student_name • cmi.core.lesson_location • cmi.core.credit • cmi.core.lesson_status • cmi.core.entry • cmi.core.score._children • cmi.core.score.raw • cmi.core.total_time • cmi.core.exit • cmi.core.session_time • cmi.suspend_data • cmi.launch_data

Req No	Requirement
2.2.2-2	If the SCO, through it's correct usage of the Run-Time Environment API, has achieved "SCORM Version 1.1 Run-Time Environment Conformant – Minimum"(SCO-RTE1) status, then in order to also be considered "SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Optional" (SCO–RTE1+Optional), the SCO shall correctly set or get one or more of the following LMS Optional data model elements: • cmi.core.score.max • cmi.core.score.min • cmi.core.lesson_mode • cmi.comments • cmi.comments_from_lms • cmi.objectives._children • cmi.objectives._count • cmi.objectives.n.id • cmi.objectives.n.score._children • cmi.objectives.n.score.raw • cmi.objectives.n.score.max • cmi.objectives.n.score.min • cmi.objectives.n.status • cmi.student_data._children • cmi.student_data.mastery_score • cmi.student_data.max_time_allowed • cmi.student_data.time_limit_action • cmi.student_preference._children • cmi.student_preference.audio • cmi.student_preference.language • cmi.student_preference.speed • cmi.student_preference.text • cmi.interactions._children • cmi.interactions._count • cmi.interactions.n.id • cmi.interactions.n.objectives._count • cmi.interactions.n.objectives.n.id • cmi.interactions.n.time • cmi.interactions.n.type • cmi.interactions.n.correct_responses._count • cmi.interactions.n.correct_responses.n.pattern • cmi.interactions.n.weighting

Req No	Requirement
	• cmi.interactions.n.student_response • cmi.interactions.n.result • cmi.interactions.n.latency
2.2.2-3	The SCO shall only attempt to set or get CMI data model elements that are supported by the SCORM Run-Time Environment Data Model.
2.2.2-3.1	The SCO shall only attempt to get (read) data model elements that are defined as "read-write" or "read-only". These elements are: • cmi.core._children (read-only) • cmi.core.student_id (read-only) • cmi.core.student_name (read-only) • cmi.core.lesson_location (read/write) • cmi.core.credit (read-only) • cmi.core.lesson_status (read/write) • cmi.core.entry (read-only) • cmi.core.score._children (read-only) • cmi.core.score.raw (read/write) • cmi.core.score.max (read/write) • cmi.core.score.min (read/write) • cmi.core.total_time (read-only) • cmi.core.lesson_mode (read-only) • cmi.suspend_data (read/write) • cmi.launch_data (read-only) • cmi.comments (read/write) • cmi.comments_from_lms (read-only) • cmi.objectives._children (read-only) • cmi.objectives._count (read-only) • cmi.objectives.n.id (read/write) • cmi.objectives.n.score._children (read-only) • cmi.objectives.n.score.raw (read/write) • cmi.objectives.n.score.max (read/write) • cmi.objectives.n.score.min (read/write) • cmi.objectives.n.status (read/write) • cmi.student_data._children (read-only) • cmi.student_data.mastery_score (read-only) • cmi.student_data.max_time_allowed (read-only) • cmi.student_data.time_limit_action (read-only) • cmi.student_preference._children (read-only) • cmi.student_preference.audio (read/write)

Req No	Requirement
	• cmi.student_preference.language (read/write) • cmi.student_preference.speed (read/write) • cmi.student_preference.text (read/write) • cmi.interactions._children (read-only) • cmi.interactions._count (read-only) • cmi.interactions.n.objectives._count (read-only) • cmi.interactions.n.correct_responses._count (read-only)
2.2.2-3.2	The SCO shall only attempt to set (write) CMI data model elements that are defined as "read-write" or "write-only". These elements are: • cmi.core.lesson_location (read/write) • cmi.core.lesson_status (read/write) • cmi.core.score.raw (read/write) • cmi.core.score.max (read/write) • cmi.core.score.min (read/write) • cmi.core.exit (write-only) • cmi.core.session_time (write-only) • cmi.suspend_data (read/write) • cmi.comments (read/write) • cmi.objectives.n.id (read/write) • cmi.objectives.n.score.raw (read/write) • cmi.objectives.n.score.max (read/write) • cmi.objectives.n.score.min (read/write) • cmi.objectives.n.status (read/write) • cmi.student_preference.audio (read/write) • cmi.student_preference.language (read/write) • cmi.student_preference.speed (read/write) • cmi.student_preference.text (read/write) • cmi.interactions.n.id (write-only) • cmi.interactions.n.objectives.n.id (write-only) • cmi.interactions.n.time (write-only) • cmi.interactions.n.type (write-only) • cmi.interactions.n.correct_responses.n.pattern (write-only) • cmi.interactions.n.weighting (write-only) • cmi.interactions.n.student_response (write-only) • cmi.interactions.n.result (write-only) • cmi.interactions.n.latency (write-only)

Req No	Requirement
2.2.2-4	When setting (writing) data model element values, the SCO shall provide the value for the element according to the data model data type and vocabulary definition (if any) for the corresponding data model element as follows: • cmi.core.lesson_location - CMIString255 • cmi.core.lesson_status - CMIVocabulary (Status) • cmi.core.score.raw - CMIDecimal or CMIBlank • cmi.core.score.max - CMIDecimal or CMIBlank • cmi.core.score.min - CMIDecimal or CMIBlank • cmi.core.exit - CMIVocabulary (Exit) • cmi.core.session_time - CMITimespan • cmi.suspend_data - CMIString4096 • cmi.comments - CMIString4096 • cmi.objectives.n.id - CMIIdentifier • cmi.objectives.n.score.raw - CMIDecimal or CMIBlank • cmi.objectives.n.score.max - CMIDecimal or CMIBlank • cmi.objectives.n.score.min - CMIDecimal or CMIBlank • cmi.objectives.n.status - CMIVocabulary (Status) • cmi.student_preference.audio - CMISInteger • cmi.student_preference.language - CMIString255 • cmi.student_preference.speed - CMISInteger • cmi.student_preference.text - CMISInteger • cmi.interactions.n.id - CMIIdentifier • cmi.interactions.n.objectives.n.id - CMIIdentifier • cmi.interactions.n.time - CMITime • cmi.interactions.n.type - CMIVocabulary (Interaction) • cmi.interactions.n.correct_responses.n.pattern - CMIFeedback • cmi.interactions.n.weighting - CMIDecimal • cmi.interactions.n.student_response - CMIFeedback • cmi.interactions.n.result - Vocabulary (Result) • cmi.interactions.n.latency - CMITimespan
2.2.2-5	***(Not tested)*** When setting (writing) the cmi.core.exit element, the SCO should set the value according to the following: • "time-out" - This indicates that the SCO ended because the SCO has determined an excessive amount of time has elapsed, or the max_time_allowed has been exceeded. The max_time_allowed can be found in the CSF (content | block | sco | timeLimit |

Req No	Requirement
	maxTimeAllowed) • "suspend" - This indicates that the student left the SCO with the intent of returning to it later at the point where he/she left. • "logout" - This indicates that the student logged out from within the SCO instead of returning to the LMS system to log out. This implies that the SCO passed control to the LMS system, and the LMS system automatically logged the student out of the course -- after updating the appropriate data model elements. • "" - The empty string should be used to represent a normal exit state.
2.2.2-6	***(Not tested)*** When setting (writing) the cmi.core.lesson_status element, the SCO shall set the value according to the following: • "passed" – This value is used when the SCO is taken for credit (i.e. cmi.core.credit is "credit"). • "failed" – This value is used when the SCO is taken for credit (i.e. cmi.core.credit is "credit"). • "completed" – This value is used when the SCO is taken for no-credit (i.e. cmi.core.credit is "no-credit"). • "incomplete" – This value is used when the SCO is taken for no-credit or credit, when the SCO is exited prematurely (before a passed/failed/completed status can be determined). • "browsed" – This value is used when the cmi.core.lesson_mode is "browse". Note: If the SCO attempts to get the value of cmi.core.lesson_mode, and the LMS returns an empty string and sets the API Error Code to "401" (not implemented), the SCO should assume a mode of "normal".

Table 2.2.2a SCO Run-Time Environment Data Model Conformance Requirements

2.3. Meta-Data Conformance Requirements

This section defines the conformance requirements for Raw Media, Content and Course Meta-data XML documents as defined in Section 2.2 of the SCORM Version 1.1.

This section contains two requirements tables. The first table, Table 2.3.1.a, contains the requirements that must be met in order for a meta-data instance to achieve certification to the 4 different meta-data conformance categories:

1. "SCORM Version 1.1 Meta-data XML Conformant – Minimum" (MD-XML1)
2. "SCORM Version 1.1 Meta-data XML Conformant – Minimum with Optional Elements" (MD-XML1+Optional)
3. "SCORM Version 1.1 Meta-data XML Conformant – Minimum with Extensions" (MD-XML1+Extensions)
4. "SCORM Version 1.1 Meta-data XML Conformant – Minimum with Optional Elements and Extensions" (MD-XML1+Optional+Extensions)

The second table, Table 2.3.1b, contains the conformance requirements for implementation of all of the meta-data elements. For example, requirement 2.3.1-2 states which meta-data elements must be implemented correctly in order for the meta-data instance to be certified "SCORM Version 1.1 Meta-data XML Conformant – Minimum with Optional Elements" (MD-XML1+Optional). The requirements in Table 2.3.1b specify how those meta-data elements are to be implemented correctly.

The requirements in the following table apply to Raw Media, Content and Course Meta-data instances as specified within the table entries.

Meta-data Elements are designated as mandatory or optional for implementation within a given type (i.e. Course, Content, Raw Media) of meta-data document. If an element is designated as mandatory, it must appear within the test subject XML meta-data instance and contain valid data. If a document is optional, it may or may not appear in the test subject meta-data instance. If the optional element does appear, it must contain valid data.

Meta-data elements are mandatory for the corresponding meta-data type if the minimum cardinality for the element is specified to be one (1). For example, requirement 2.3.1-4.3 in table 2.3.1b states that both Content and Course meta-data instances shall contain one (1) or more (many) <catalogentry> element tags. This indicates that the minimum cardinality is one (1) and therefore, this element is mandatory for Content and Course meta-data.

Elements are optional if the specified minimum cardinality for an element is 0. For example, in requirement 2.3.1-6.2 in table 2.3.1b states that Raw Media meta-data has zero (0) or more <catalogentry> elements. The minimum cardinality in this case is zero

therefore indicating that this element is optional for Raw Media meta-data. It should also be noted that the scope of the mandatory/optional implementation for a particular element is defined as the parent element.

The requirements shown in table 2.3.1b also contain several "best practice" requirements. Failure to implement these "best practice" requirements within a meta-data instance does not, by itself, prevent the meta-data instance from achieving conformance. Failure to adhere to a "best practice" requirement results in a warning message being generated during the conformance testing process.

"Best practice" requirements are noted as such in table 2.3.1b. For example, requirement 2.3.1-4.3 includes the statement for the <catalogentry> element: "Up to 8 supported – Not a conformance check, warning only". If the test subject meta-data document contains more than 8 <catalogentry> elements (within a single <general> element) then the document will not be deemed to be non-conformant for this reason.

Note that the following requirements table (table 2.3.1a) does specifically address requirements for achieving SCORM Version 1.1 Meta-data XML Conformant – Minimum with Optional Elements and Extensions" (MD-XML1+Optional+Extensions). A meta-data instance is determined to be SCORM Version 1.1 Meta-data XML Conformant – Minimum with Optional Elements and Extensions" (MD-XML1+Optional+Extensions) if it correctly implements the requirements for SCORM Version 1.1 Meta-data XML Conformant – Minimum with Optional Elements" (MD-XML1+Optional) and the requirements for SCORM Version 1.1 Meta-data XML Conformant – Minimum with Extensions" (MD-XML1+Extensions).

Each requirement that deals with specific meta-data elements, where appropriate, references the element number as it appears in the SCORM Version 1.1 and in the IEEE Learning Technology Standards Committee (LTSC) Learning Objects Metadata (LOM) Specification[9]. This is referred to in the following tables as the LOM Element Number.

<table>
<tr><th>Req No</th><th>Requirement</th></tr>
<tr><td>2.3.1-1</td><td>In order to be "Meta-data XML Conformant – Minimum" (MD–XML1):</td></tr>
<tr><td>2.3.1-1.1</td><td>A Content (SCO/ Block) or Course (Content Aggregation) meta-data instance shall contain the following mandatory elements:
<table>
<tr><th>LOM Element Number</th><th>Element</th></tr>
<tr><td>1</td><td>general</td></tr>
<tr><td>1.2</td><td>title</td></tr>
<tr><td>1.3</td><td>catalogentry</td></tr>
<tr><td>1.3.1</td><td>catalogue</td></tr>
</table>
</td></tr>
</table>

Req No	Requirement
	1.3.2 entry 1.5 description 1.6 keywords 2 lifecycle 2.1 version 2.2 status 3 metametadata 3.4 metadatascheme 4 technical 4.1 format 4.3 location 6 rights 6.1 cost 6.2 copyrightandotherrestrictions 9 classification 9.1 purpose 9.3 description 9.4 keywords
2.3.1-1.2	A Raw Media (Asset) meta-data instance shall contain the following mandatory elements: LOM: Element 1 general 1.2 title 1.5 description 3 metametadata 3.4 metadatascheme 4 technical 4.1 format 4.3 location 6 rights 6.1 cost 6.2 copyrightandotherrestrictions

<table>
<tr><th>Req No</th><th>Requirement</th></tr>
<tr><td></td><td><table><tr><td>6.2</td><td>copyrightandotherrestrictions</td></tr></table></td></tr>
<tr><td>2.3.1-2</td><td>In order to be “Meta-data XML Conformant – Minimum with Optional Elements” (MD–XML1+Optional):</td></tr>
<tr><td>2.3.1-2.1</td><td>A Content (SCO/ Block) or Course (Content Aggregation) meta-data instance shall contain one or more of the following elements:<table><tr><th>LOM</th><th>Element</th></tr><tr><td>1.4</td><td>language</td></tr><tr><td>1.7</td><td>coverage</td></tr><tr><td>1.8</td><td>structure</td></tr><tr><td>1.9</td><td>aggregationlevel</td></tr><tr><td>2.3</td><td>contribute</td></tr><tr><td>2.3.1</td><td>role</td></tr><tr><td>2.3.2</td><td>centity</td></tr><tr><td>2.3.3</td><td>date</td></tr><tr><td>3.2</td><td>catalogentry</td></tr><tr><td>3.2.1.</td><td>catalogue</td></tr><tr><td>3.2.2</td><td>entry</td></tr><tr><td>3.3</td><td>contribute</td></tr><tr><td>3.3.2</td><td>role</td></tr><tr><td>3.3.2</td><td>centity</td></tr><tr><td>3.3.3</td><td>date</td></tr><tr><td>3.5</td><td>language</td></tr><tr><td>4.2</td><td>size</td></tr><tr><td>4.4</td><td>requirements</td></tr><tr><td>4.4.1</td><td>type</td></tr><tr><td>4.4.2</td><td>name</td></tr><tr><td>4.4.3</td><td>minimumversion</td></tr><tr><td>4.4.4</td><td>maximumversion</td></tr><tr><td>4.5</td><td>installationremarks</td></tr><tr><td>4.6</td><td>otherplatformrequirements</td></tr><tr><td>4.7</td><td>duration</td></tr><tr><td>5</td><td>educational</td></tr><tr><td>5.1</td><td>interactivitytype</td></tr></table></td></tr>
</table>

<table>
<tr><th>Req No</th><th>Requirement</th></tr>
<tr><td></td><td>
<table>
<tr><td>5.1</td><td>interactivitytype</td></tr>
<tr><td>5.2</td><td>learningresourcetype</td></tr>
<tr><td>5.3</td><td>interactivitylevel</td></tr>
<tr><td>5.4</td><td>semanticdensity</td></tr>
<tr><td>5.5</td><td>intendedenduserrole</td></tr>
<tr><td>5.6</td><td>learningcontext</td></tr>
<tr><td>5.7</td><td>typicalagerange</td></tr>
<tr><td>5.8</td><td>difficulty</td></tr>
<tr><td>5.9</td><td>typicallearningtime</td></tr>
<tr><td>5.10</td><td>description</td></tr>
<tr><td>5.11</td><td>language</td></tr>
<tr><td>6.3</td><td>description</td></tr>
<tr><td>7</td><td>relation</td></tr>
<tr><td>7.1</td><td>kind</td></tr>
<tr><td>7.2</td><td>resource</td></tr>
<tr><td>7.2.2</td><td>description</td></tr>
<tr><td>8</td><td>annotation</td></tr>
<tr><td>8.1</td><td>centity</td></tr>
<tr><td>8.2</td><td>date</td></tr>
<tr><td>8.3</td><td>description</td></tr>
<tr><td>9.2</td><td>taxonpath</td></tr>
<tr><td>9.2.1</td><td>source</td></tr>
<tr><td>9.2.2</td><td>taxon</td></tr>
<tr><td>9.2.2.1</td><td>id</td></tr>
<tr><td>9.2.2.2</td><td>entry</td></tr>
</table>
</td></tr>
<tr><td>2.3.1-2.2</td><td>A Raw Media (Asset) meta-data instance shall contain one or more of the following elements:
<table>
<tr><th>LOM</th><th>Element</th></tr>
<tr><td>1.3</td><td>catalogentry</td></tr>
<tr><td>1.3.1</td><td>catalogue</td></tr>
<tr><td>1.3.2</td><td>entry</td></tr>
<tr><td>1.4</td><td>language</td></tr>
<tr><td>1.6</td><td>keywords</td></tr>
<tr><td>1.7</td><td>coverage</td></tr>
</table>
</td></tr>
</table>

Req No	Requirement			
		1.7	coverage	
		1.8	structure	
		1.9	aggregationlevel	
		2	lifecycle	
		2.1	version	
		2.2	status	
		2.3	contribute	
		2.3.1	role	
		2.3.2	centity	
		2.3.3	date	
		3.2	catalogentry	
		3.2.1.	catalogue	
		3.2.2	entry	
		3.3	contribute	
		3.3.1	role	
		3.3.2	centity	
		3.3.3	date	
		3.5	language	
		4.2	size	
		4.4	requirements	
		4.4.1	type	
		4.4.2	name	
		4.4.3	minimumversion	
		4.4.4	maximumversion	
		4.5	installationremarks	
		4.6	otherplatformrequirements	
		4.7	duration	
		5	educational	
		5.1	interactivitytype	
		5.2	learningresourcetype	
		5.3	interactivitylevel	
		5.4	semanticdensity	
		5.5	intendedenduserrole	
		5.6	learningcontext	
		5.7	typicalagerange	
		5.8	difficulty	

<table>
<tr><th>Req No</th><th>Requirement</th></tr>
<tr><td></td><td>
<table>
<tr><td>5.9</td><td>typicallearningtime</td></tr>
<tr><td>5.10</td><td>description</td></tr>
<tr><td>5.11</td><td>language</td></tr>
<tr><td>6.3</td><td>description</td></tr>
<tr><td>7</td><td>relation</td></tr>
<tr><td>7.1</td><td>kind</td></tr>
<tr><td>7.2</td><td>resource</td></tr>
<tr><td>7.2.2</td><td>description</td></tr>
<tr><td>8</td><td>annotation</td></tr>
<tr><td>8.1</td><td>centity</td></tr>
<tr><td>8.2</td><td>date</td></tr>
<tr><td>8.3</td><td>description</td></tr>
<tr><td>9</td><td>classification</td></tr>
<tr><td>9.1</td><td>purpose</td></tr>
<tr><td>9.2</td><td>taxonpath</td></tr>
<tr><td>9.2.1</td><td>source</td></tr>
<tr><td>9.2.2</td><td>taxon</td></tr>
<tr><td>9.2.2.1</td><td>id</td></tr>
<tr><td>9.2.2.2</td><td>entry</td></tr>
<tr><td>9.3</td><td>description</td></tr>
<tr><td>9.4</td><td>keywords</td></tr>
</table>
</td></tr>
<tr><td>2.3.1-3</td><td>In order to be “SCORM Version 1.1 Meta-data XML Conformant – Minimum with Extensions” (MD–XML1+Extensions) A Raw Media (Asset), Content (SCO/ Block) or Course (Content Aggregation) meta-data instance shall contain one or more valid <extension> elements.</td></tr>
<tr><td>2.3.1-3.1</td><td>All extension elements (<extension>) shall be valid according to the IMS Learning Resource Meta-Data DTD[10].</td></tr>
</table>

Table 2.3.1a – Meta-Data Conformance Requirements by Conformance Category

The following, Table 2.3.1b, contains the SCORM Conformance requirements for conformant implementation of all of the meta-data elements.

Req No	Requirement
2.3.1-4	The Meta-data XML document shall contain a **<general>** element (LOM element 1) that adheres to the following: Raw Media (Asset) • 1 and only 1 element tag <general> Content (SCO/ Block) / Course (Content Aggregation) • 1 and only 1 element tag <general>
2.3.1-4.1	The Meta-data XML document shall NOT contain an **<identifier>** element (LOM element 1.1) This element is reserved for future use and has therefore been removed from the IMS Learning Resource Meta-Data DTD[10].
2.3.1-4.2	The Meta-data XML document shall contain a **<title>** element (LOM element 1.2) that adheres to the following: Raw Media (Asset) • 1 and only 1 element tag <title> • Maximum of 1024 characters • Bound within 1 or more <langstring> elements Content (SCO/ Block) / Course (Content Aggregation) • 1 and only 1 element tag <title> • Maximum of 1024 characters • Bound within 1 or more <langstring> elements
2.3.1-4.3	The Meta-data XML document shall contain a **<catalogentry>** element (LOM element 1.3) that adheres to the following: Raw Media (Asset) • 0 or more element tag <catalogentry> • Up to 8 supported – Not a conformance check, warning only Content (SCO/ Block) / Course (Content Aggregation) • 1 or more element tag <catalogentry> • Up to 8 supported – Not a conformance check, warning only
2.3.1-4.3.1	The Meta-data XML document shall contain a **<catalogue>** element (LOM element 1.3.1) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <catalogue> • Maximum of 1024 characters Content (SCO/ Block) / Course (Content Aggregation) • 1 and only 1 element tag <catalogue> • Maximum of 1024 characters

Req No	Requirement
2.3.1-4.3.2	The Meta-data XML document shall contain a **<entry>** element (LOM element 1.3.2) that adheres to the following: Raw Media (Asset) • 0 or 1 • Maximum of 1024 characters • Bound within 1 or more <langstring> elements Content (SCO/ Block) / Course (Content Aggregation) • 1 and only 1 element tag <entry> • Maximum of 1024 characters • Bound within 1 or more <langstring> elements
2.3.1-4.4	The Meta-data XML document shall contain a **<language>** element (LOM element 1.4) that adheres to the following: Raw Media (Asset) • 0 or more element tag <language> • Up to 8 supported – Not a conformance check, warning only • Maximum of 128 characters • *Expressed as per ISO 639*[11] *and ISO 3166*[12] Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tag <language> • Up to 8 supported – Not a conformance check, warning only • Maximum of 128 characters • *Expressed as per ISO 639*[11] *and ISO 3166*[12]
2.3.1-4.5	The Meta-data XML document shall contain a **<description>** element (LOM element 1.5) that adheres to the following: Raw Media (Asset) • 1 or more element tag <description> • Up to 8 supported – Not a conformance check, warning only • Maximum of 2048 characters • Bound within 1 or more <langstring> element Content (SCO/ Block) / Course (Content Aggregation) • 1 or more element tag <description> • Up to 8 supported – Not a conformance check, warning only • Maximum of 2048 characters • Bound within 1 or more <langstring> element
2.3.1-4.6	The Meta-data XML document shall contain a **<keywords>** element (LOM element 1.6) that adheres to the following: Raw Media (Asset) • 0 or more element tags <keywords> • Up to 8 supported – Not a conformance check, warning only • Maximum of 1024 characters • Bound within 1 or more <langstring> elements Content (SCO/ Block) / Course (Content Aggregation)

Req No	Requirement
	• 1 or more element tags <keywords> • Up to 8 supported – Not a conformance check, warning only • Maximum of 1024 characters • Bound within 1 or more <langstring> elements
2.3.1-4.7	The Meta-data XML document shall contain a **<coverage>** element (LOM element 1.7) that adheres to the following: Raw Media (Asset) • 0 or more element tags <coverage> • Up to 8 supported – Not a conformance check, warning only • Maximum of 1024 characters • Bound within 1 or more <langstring> elements Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tags <coverage> • Up to 8 supported – Not a conformance check, warning only • Maximum of 1024 characters • Bound within 1 or more <langstring> elements

2.3.1-4.8	The Meta-data XML document shall contain a **<structure>** element (LOM element 1.8) that adheres to the following:
	Raw Media (Asset)
	• 0 or 1 element tag <structure>
	• Maximum of 32 characters
	• Bound within 1 and only 1 <langstring> elements
	• Member of the following Restricted Vocabulary list
	  ○ User_defined
	  ○ See_classification
	  ○ Collection
	  ○ Mixed
	  ○ Linear
	  ○ Hierarchical
	  ○ Networked
	  ○ Branched
	  ○ Parceled
	  ○ Atomic
	Content (SCO/ Block) / Course (Content Aggregation)
	• 0 or 1 element tag <structure>
	• Maximum of 32 characters
	• Bound within 1 and only 1 <langstring> elements
	• Member of the following Restricted Vocabulary list
	  ○ User_defined
	  ○ See_classification
	  ○ Collection
	  ○ Mixed

Req No	Requirement
	○ Linear ○ Hierarchical ○ Networked ○ Branched ○ Parceled ○ Atomic
2.3.1-4.9	The Meta-data XML document shall contain a **<aggregationlevel>** element (LOM element 1.9) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <aggregationlevel> • Maximum of 8 characters • Member of the following Restricted Vocabulary list ○ 0 ○ 1 ○ 2 ○ 3 Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <aggregationlevel> • Maximum of 8 characters • Member of the following Restricted Vocabulary list ○ 0 ○ 1 ○ 2 ○ 3
2.3.1-5	The Meta-data XML document shall contain a **<lifecycle>** element (LOM element 2) that adheres to the following: Raw Media (Asset) • 0 or 1 element tags <lifecycle> Content (SCO/ Block) / Course (Content Aggregation) • 1 and only 1 element tag <lifecycle>
2.3.1-5.1	The Meta-data XML document shall contain a **<version>** element (LOM element 2.1) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <version> • Maximum of 64 characters • Bound within a <langstring> element Content (SCO/ Block) / Course (Content Aggregation) • 1 and only 1 element tag <version> • Maximum of 64 characters • Bound within a <langstring> element

Req No	Requirement

2.3.1-5.2	The Meta-data XML document shall contain a **<status>** element (LOM element 2.2) that adheres to the following:
	Raw Media (Asset)
	• 0 or 1 element tag <status>
	• Maximum of 64 characters
	• Bound within 1 and only 1 <langstring> element
	• Member of the following Restricted Vocabulary list
	  ○ User_defined
	  ○ See_classification
	  ○ Draft
	  ○ Final
	  ○ Revised
	  ○ Unavailable
	Content (SCO/ Block) / Course (Content Aggregation)
	• 1 and only 1 element tag <status>
	• Maximum of 64 characters
	• Bound within 1 and only 1 <langstring> element
	• Member of the following Restricted Vocabulary list
	  ○ User_defined
	  ○ See_classification
	  ○ Draft
	  ○ Final
	  ○ Revised
	  ○ Unavailable

2.3.1-5.3	The Meta-data XML document shall contain a **<contribute>** element (LOM element 2.3) that adheres to the following: Raw Media (Asset) • 0 or more element tag <contribute> • Up to 32 supported – Not a conformance check, warning only Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tag <contribute> • Up to 32 supported – Not a conformance check, warning only

2.3.1-5.3.1	The Meta-data XML document shall contain a **<role>** element (LOM element 2.3.1) that adheres to the following:
	Raw Media (Asset)
	• 0 or 1 element tag <role>
	• Bound within 1 and only 1 <langstring> element
	• Checked against Best Practice vocabulary list
	  ○ User_defined
	  ○ See_classification
	  ○ Author (Recommended that exactly 1 Author exist)
	  ○ Publisher
	  ○ Unknown

Req No	Requirement
	○ Initiator ○ Terminator ○ Validator ○ Editor ○ Graphical Designer ○ Technical Implementer ○ Content Provider ○ Technical Validator ○ Educational Validator ○ Script Writer ○ Instructional Designer • If not in the Best Practice list, Maximum of 128 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <role> • Bound within 1 and only 1 <langstring> element • Checked against Best Practice vocabulary list ○ User_defined ○ See_classification ○ Author (Recommended that exactly 1 Author exist) ○ Publisher ○ Unknown ○ Initiator ○ Terminator ○ Validator ○ Editor ○ Graphical Designer ○ Technical Implementer ○ Content Provider ○ Technical Validator ○ Educational Validator ○ Script Writer ○ Instructional Designer • If not in the Best Practice list, Maximum of 128 characters
2.3.1-5.3.2	The Meta-data XML document shall contain a **<centity>** element (LOM element 2.3.2) that adheres to the following: Raw Media (Asset) • 0 or more element tags <centity> • Up to 8 supported – Not a conformance check, warning only • Maximum of 1024 characters • Bound within a <vcard> element Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tags <centity> • Up to 8 supported – Not a conformance check, warning only

Req No	Requirement
	• Maximum of 1024 characters • Bound within a <vcard> element

2.3.1-5.3.3	The Meta-data XML document shall contain a **<date>** element (LOM element 2.3.3) that adheres to the following:
	Raw Media (Asset)
	• 0 or 1 element tags <date>
	• Must be bound in a Date Type
	  ○ <datetime> element must be expressed as per ISO 8601[14]
	  ○ <datetime> element best practice: Universal date/time format
	  ○ <description> element
	  ▪ must be bound within a <langstring> element
	  ▪ Maximum of 2048 characters
	Content (SCO/ Block) / Course (Content Aggregation)
	• 0 or 1 element tags <date>
	• Must be bound in a Date Type
	  ○ <datetime> element must be expressed as per ISO 8601[14]
	  ○ <datetime> element best practice: Universal date/time format
	  ○ <description> element
	  ▪ Must be bound within a <langstring> element.
	  ▪ Maximum of 2048 characters

2.3.1-6	The Meta-data XML document shall contain a **<metametadata>** element (LOM element 3) that adheres to the following: Raw Media (Asset) • 1 and only 1 element tag <metametadata> Content (SCO/ Block) / Course (Content Aggregation) • 1 and only 1 element tag <metametadata>
2.3.1-6.1	The Meta-data XML document shall contain a **<identifier>** element (LOM element 3.1) that adheres to the following: Raw Media (Asset) • this element is RESERVED and should not be used Content (SCO/ Block) / Course (Content Aggregation) • this element is RESERVED and should not be used
2.3.1-6.2	The Meta-data XML document shall contain a **<catalogentry>** element (LOM element 3.2) that adheres to the following: Raw Media (Asset) • 0 or more element tags <catalogentry> • Up to 8 supported – Not a conformance check, warning only Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tags <catalogentry> • Up to 8 supported – Not a conformance check, warning only

Req No	Requirement
2.3.1-6.2.1	The Meta-data XML document shall contain a **<catalogue>** element (LOM element 3.2.1) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <catalogue> • Maximum of 1024 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <catalogue> • Maximum of 1024 characters
2.3.1-6.2.2	The Meta-data XML document shall contain a **<entry>** element (LOM element 3.2.2) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <entry> • Maximum of 1024 characters • Bound within 1 or more <langstring> elements Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <entry> • Maximum of 1024 characters • Bound within 1 or more <langstring> elements
2.3.1-6.3	The Meta-data XML document shall contain a **<contribute>** element (LOM element 3.3) that adheres to the following: Raw Media (Asset) • 0 or more element tag <contribute> • Up to 8 supported – Not a conformance check, warning only Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tag <contribute> • Up to 8 supported – Not a conformance check, warning only

2.3.1-6.3.1	The Meta-data XML document shall contain a **<role>** element (LOM element 3.3.1) that adheres to the following:
	Raw Media (Asset)
	• 0 or 1 element tag <role>
	• Bound within 1 and only 1 <langstring> element
	• Checked against Best Practice vocabulary list
	  ○ User_defined
	  ○ See_classification
	  ○ Creator (Recommended that exactly 1 Creator exist)
	  ○ Validator
	• If not in the Best Practice list, Maximum of 128 characters
	Content (SCO/ Block) / Course (Content Aggregation)
	• 0 or 1 element tag <role>
	• Bound within 1 and only 1 <langstring> element
	• Checked against Best Practice vocabulary list
	  ○ User_defined

Req No	Requirement
	○ See_classification ○ Creator (Recommended that exactly 1 Creator exist) ○ Validator • If not in the Best Practice list, Maximum of 128 characters
2.3.1-6.3.2	The Meta-data XML document shall contain a **<centity>** element (LOM element 3.3.2) that adheres to the following: Raw Media (Asset) • 0 or more element tags <centity> • Up to 8 supported – Not a conformance check, warning only • Maximum of 1024 characters • Bound within a <vcard> element Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tags <centity> • Up to 8 supported – Not a conformance check, warning only • Maximum of 1024 characters • Bound within a <vcard> element
2.3.1-6.3.3	The Meta-data XML document shall contain a **<date>** element (LOM element 3.3.3) that adheres to the following: Raw Media (Asset) • 0 or 1 element tags <date> • Must be bound in a Date Type ○ <datetime> element must be expressed as per ISO 8601[14] ○ <datetime> element best practice: Universal date/time format ○ <description> element ▪ Must be bound within a <langstring> element ▪ Maximum of 2048 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tags <date> • Must be bound in a Date Type ○ <datetime> element must be expressed as per ISO 8601[14] ○ <datetime> element best practice: Universal date/time format ○ <description> element ▪ Must be bound within a <langstring> element ▪ Maximum of 2048 characters
2.3.1-6.4	The Meta-data XML document shall contain a **<metadatascheme>** element (LOM element 3.4) that adheres to the following: Raw Media (Asset) • 1 or more element tags <metadatascheme> • Up to 8 supported – Not a conformance check, warning only • Maximum of 32 characters Content (SCO/ Block) / Course (Content Aggregation)

Req No	Requirement
	• 1 or more element tags <metadatascheme> • Up to 8 supported – Not a conformance check, warning only • Maximum of 32 characters
2.3.1-6.5	The Meta-data XML document shall contain a **<language>** element (LOM element 3.5) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <language> • Maximum of 128 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <language> • Maximum of 128 characters
2.3.1-7	The Meta-data XML document shall contain a **<technical>** element (LOM element 4) that adheres to the following: Raw Media (Asset) • 1 and only 1 element tag <technical> Content (SCO/ Block) / Course (Content Aggregation) • 1 and only 1 element tag <technical>

2.3.1-7.1	The Meta-data XML document shall contain a **<format>** element (LOM element 4.1) that adheres to the following:
	Raw Media (Asset)
	• 1 and only 1 element tag <format>
	• Member of a list of Restricted Vocabulary
	  ○ Valid MIME type
	  ○ Non-digital
	• Bound within 1 or more <langstring> elements
	• Maximum of 512 characters
	Content (SCO/ Block) / Course (Content Aggregation)
	• 1 and only 1 element tag <format>
	• Member of a list of Restricted Vocabulary
	  ○ Valid MIME type
	  ○ Non-digital
	• Bound within 1 or more <langstring> elements
	• Maximum of 512 characters

Req No	Requirement

2.3.1-7.2	The Meta-data XML document shall contain a **<size>** element (LOM element 4.2) that adheres to the following:
	Raw Media (Asset)
	• 0 or 1 element tag <size>
	• Maximum of 32 characters
	  ○ Must be a valid number – represents bytes
	Content (SCO/ Block) / Course (Content Aggregation)
	• 0 or 1 element tag <size>
	• Maximum of 32 characters
	  ○ Must be a valid number – represents bytes

2.3.1-7.3	The Meta-data XML document shall contain a **<location>** element (LOM element 4.3) that adheres to the following: Raw Media (Asset) • 1 or more element tags <location> • Up to 8 supported – Not a conformance check, warning only • Maximum of 1024 characters Content (SCO/ Block) / Course (Content Aggregation) • 1 or more element tags <location> • Up to 8 supported – Not a conformance check, warning only • Maximum of 1024 characters
2.3.1-7.4	The Meta-data XML document shall contain a **<requirements>** element (LOM element 4.4) that adheres to the following: Raw Media (Asset) • 0 or more element tags <requirements> • Up to 8 supported – Not a conformance check, warning only Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tags <requirements> • Up to 8 supported – Not a conformance check, warning only

2.3.1-7.4.1	The Meta-data XML document shall contain a **<type>** element (LOM element 4.4.1) that adheres to the following:
	Raw Media (Asset)
	• 0 or 1 element tag <type>
	• Bound within 1 and only 1 <langstring> element
	• Checked against Best Practice vocabulary list
	  ○ User_defined
	  ○ See_classification
	  ○ Operating System
	  ○ Browser
	• If not in the Best Practice list, Maximum of 32 characters
	Content (SCO/ Block) / Course (Content Aggregation)
	• 0 or 1 element tag <type>
	• Bound within 1 and only 1 <langstring> element

Req No	Requirement

	• Checked against Best Practice vocabulary list
	  ○ User_defined
	  ○ See_classification
	  ○ Operating System
	  ○ Browser
	• If not in the Best Practice list, Maximum of 32 characters

2.3.1-7.4.2	The Meta-data XML document shall contain a **<name>** element (LOM element 4.4.2) that adheres to the following:
	Raw Media (Asset)
	• 0 or 1 element tag <name>
	• Bound within 1 and only 1 <langstring> element
	• Checked against Best Practice Vocabulary List
	  ○ User_defined
	  ○ See_classification
	  ○ If element 4.4.1 <type> - "Operating System"
	  ▪ PC-DOS
	  ▪ MS-Windows
	  ▪ MacOS
	  ▪ Unix
	  ▪ Multi-OS
	  ▪ Other
	  ▪ None
	  ○ If element 4.4.1 <type> - "Browser"
	  ▪ Any
	  ▪ Netscape Communicator
	  ▪ Microsoft Internet Explorer
	  ▪ Opera
	  ○ If element 4.4.1 <type> - is something other than the Best Practice Vocabulary
	  ▪ Maximum of 1024 characters
	Content (SCO/ Block) / Course (Content Aggregation)
	• 0 or 1 element tag <name>
	• Bound within 1 and only 1 <langstring> element
	• Checked against Best Practice Vocabulary List
	  ○ User_defined
	  ○ See_classification
	  ○ If element 4.4.1 <type> - "Operating System"
	  ▪ PC-DOS
	  ▪ MS-Windows
	  ▪ MacOS
	  ▪ Unix
	  ▪ Multi-OS
	  ▪ Other

Req No	Requirement
	▪ None o If element 4.4.1 <type> - "Browser" ▪ Any ▪ Netscape Communicator ▪ Microsoft Internet Explorer ▪ Opera o If element 4.4.1 <type> - is something other than the Best Practice Vocabulary ▪ Maximum of 1024 characters
2.3.1-7.4.3	The Meta-data XML document shall contain a **<minimumversion>** element (LOM element 4.4.3) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <minimumversion> • Maximum of 32 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <minimumversion> • Maximum of 32 characters
2.3.1-7.4.4	The Meta-data XML document shall contain a **<maximumversion>** element (LOM element 4.4.4) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <maximumversion> • Maximum of 32 characters • Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <maximumversion> • Maximum of 32 characters
2.3.1-7.5	The Meta-data XML document shall contain a **<installationremarks>** element (LOM element 4.5) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <installationremarks> • Maximum of 1024 characters • Bound within a <langstring> element Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <installationremarks> • Maximum of 1024 characters • Bound within a <langstring> element
2.3.1-7.6	The Meta-data XML document shall contain a **<otherplatformrequirements>** element (LOM element 4.6) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <otherplatformrequirements> • Maximum of 1024 characters • Bound within a <langstring> element

Req No	Requirement
	Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <otherplatformrequirements> • Maximum of 1024 characters • Bound within a <langstring> element

2.3.1-7.7	The Meta-data XML document shall contain a **<duration>** element (LOM element 4.7) that adheres to the following:
	Raw Media (Asset)
	• 0 or 1 element tags <duration>
	• Must be bound in a Date Type
	  ○ <datetime> element must be expressed as per ISO 8601[14]
	  ○ <datetime> element best practice: Universal date/time format
	  ○ <description> element
	  ▪ Must be bound within a <langstring> element
	  ▪ Maximum of 2048 characters
	Content (SCO/ Block) / Course (Content Aggregation)
	• 0 or 1 element tags <duration>
	• Must be bound in a Date Type
	  ○ <datetime> element must be expressed as per ISO 8601[14]
	  ○ <datetime> element best practice: Universal date/time format
	  ○ <description> element
	  ▪ Must be bound within a <langstring> element
	  ▪ Maximum of 2048 characters

2.3.1-8	The Meta-data XML document shall contain a **<educational>** element (LOM element 5) that adheres to the following: Raw Media (Asset) • 0 or 1 element tags <educational> Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tags <educational>

2.3.1-8.1	The Meta-data XML document shall contain a **<interactivitytype>** element (LOM element 5.1) that adheres to the following:
	Raw Media (Asset)
	• 0 or 1 element tag <interactivitytype>
	• Maximum of 32 characters
	• Bound within 1 and only 1 <langstring> elements
	• Member of the following Restricted Vocabulary list
	  ○ User_defined
	  ○ See_classification
	  ○ Active
	  ○ Expositive
	  ○ Mixed
	  ○ Undefined

Req No	Requirement

	Content (SCO/ Block) / Course (Content Aggregation)
	• 0 or 1 element tag <interactivitytype>
	• Maximum of 32 characters
	• Bound within 1 and only 1 <langstring> elements
	• Member of the following Restricted Vocabulary list
	  ○ User_defined
	  ○ See_classification
	  ○ Active
	  ○ Expositive
	  ○ Mixed
	  ○ Undefined

2.3.1-8.2	The Meta-data XML document shall contain a **<learningresourcetype>** element (LOM element 5.2) that adheres to the following:
	Raw Media (Asset)
	• 0 or more element tags <learningresourcetype>
	• Up to 8 supported – Not a conformance check, warning only
	• Bound within 1 and only 1 <langstring> element
	• Checked against Best Practice Vocabulary list
	  ○ User_defined
	  ○ See_classification
	  ○ Exercise
	  ○ Simulation
	  ○ Questionnaire
	  ○ Diagram
	  ○ Figure
	  ○ Graph
	  ○ Index
	  ○ Slide
	  ○ Table
	  ○ Narrative Text
	  ○ Exam
	  ○ Experiment
	  ○ ProblemStatement
	  ○ SelfAssesment
	• If not in the Best Practice list, Maximum of 1024 characters
	Content (SCO/ Block) / Course (Content Aggregation)
	• 0 or more element tags <learningresourcetype>
	• Up to 8 supported – Not a conformance check, warning only
	• Bound within 1 and only 1 <langstring> element
	• Checked against Best Practice Vocabulary list
	  ○ User_defined
	  ○ See_classification
	  ○ Exercise

Req No	Requirement
	○ Simulation ○ Questionnaire ○ Diagram ○ Figure ○ Graph ○ Index ○ Slide ○ Table ○ Narrative Text ○ Exam ○ Experiment ○ ProblemStatement ○ SelfAssesment • If not in the Best Practice list, Maximum of 1024 characters
2.3.1-8.3	The Meta-data XML document shall contain a **<interactivitylevel>** element (LOM element 5.3) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <interactivitylevel> • Member of the following Restricted Vocabulary list ○ 0 ○ 1 ○ 2 ○ 3 ○ 4 • Maximum of 8 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <interactivitylevel> • Member of the following Restricted Vocabulary list ○ 0 ○ 1 ○ 2 ○ 3 ○ 4 • Maximum of 8 characters
2.3.1-8.4	The Meta-data XML document shall contain a **<semanticdensity>** element (LOM element 5.4) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <semanticdensity> • Member of the following Restricted Vocabulary list ○ 0 ○ 1 ○ 2 ○ 3

Req No	Requirement
	○ 4 • Maximum of 8 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <semanticdensity> • Member of the following Restricted Vocabulary list ○ 0 ○ 1 ○ 2 ○ 3 ○ 4 • Maximum of 8 characters
2.3.1-8.5	The Meta-data XML document shall contain a **<intendedenduserrole>** element (LOM element 5.5) that adheres to the following: Raw Media (Asset) • 0 or more element tags <intendedenduserrole> • Up to 4 supported – Not a conformance check, warning only • Bound within 1 and only 1 <langstring> elements • Member of the following Restricted Vocabulary list ○ Teacher ○ Author ○ Learner ○ Manager • Maximum of 32 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tags <intendedenduserrole> • Up to 4 supported – Not a conformance check, warning only • Bound within 1 and only 1 <langstring> elements • Member of the following Restricted Vocabulary list ○ Teacher ○ Author ○ Learner ○ Manager • Maximum of 32 characters
2.3.1-8.6	The Meta-data XML document shall contain a **<learningcontext>** element (LOM element 5.6) that adheres to the following: Raw Media (Asset) • 0 or more element tags <learningcontext> • Up to 8 supported – Not a conformance check, warning only • Bound within 1 and only 1 <langstring> element • Checked against the Best Practice Vocabulary list ○ User_defined ○ See_classification ○ Primary Education

Req No	Requirement
	○ Secondary Education ○ Higher Education ○ University First Cycle ○ University Second Cycle ○ University Postgrade ○ Technical School First Cycle ○ Technical School Second Cycle ○ Professional Formation ○ Continuous Formation ○ Vocational Training ○ Other • If not part of the Best Practice Vocabulary list, Maximum of 128 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tags <learningcontext> • Up to 8 supported – Not a conformance check, warning only • Bound within 1 and only 1 <langstring> element • Checked against the Best Practice Vocabulary list ○ User_defined ○ See_classification ○ Primary Education ○ Secondary Education ○ Higher Education ○ University First Cycle ○ University Second Cycle ○ University Postgrade ○ Technical School First Cycle ○ Technical School Second Cycle ○ Professional Formation ○ Continuous Formation ○ Vocational Training ○ Other • If not part of the Best Practice Vocabulary list, Maximum of 128 characters
2.3.1-8.7	The Meta-data XML document shall contain a **<typicalagerange>** element (LOM element 5.7) that adheres to the following: Raw Media (Asset) • 0 or more element tags <typicalagerange> • Up to 4 supported – Not a conformance check, warning only • Bound within a <langstring> element • Maximum of 1024 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tags <typicalagerange>

Req No	Requirement
	• Up to 4 supported – Not a conformance check, warning only • Bound within a <langstring> element • Maximum of 1024 characters

2.3.1-8.8	The Meta-data XML document shall contain a **<difficulty>** element (LOM element 5.8) that adheres to the following:
	Raw Media (Asset)
	• 0 or 1 element tag <difficulty>
	• Member of the following Restricted Vocabulary list
	  ○ 0
	  ○ 1
	  ○ 2
	  ○ 3
	  ○ 4
	• Maximum of 8 characters
	Content (SCO/ Block) / Course (Content Aggregation)
	• 0 or 1 element tag <difficulty>
	• Member of the following Restricted Vocabulary list
	  ○ 0
	  ○ 1
	  ○ 2
	  ○ 3
	  ○ 4
	• Maximum of 8 characters

2.3.1-8.9	The Meta-data XML document shall contain a **<typicallearningtime>** element (LOM element 5.9) that adheres to the following:
	Raw Media (Asset)
	• 0 or 1 element tag <typicallearningtime>
	• Must be bound in a Date Type
	  ○ <datetime> element must be expressed as per ISO 8601[14]
	  ○ <datetime> element best practice: Universal date/time format
	  ○ <description> element
	  ▪ Must be bound within a <langstring> element
	  ▪ Maximum of 2048 characters
	Content (SCO/ Block) / Course (Content Aggregation)
	• 0 or 1 element tag <typicallearningtime>
	• Must be bound in a Date Type
	  ○ <datetime> element must be expressed as per ISO 8601[14]
	  ○ <datetime> element best practice: Universal date/time format
	  ○ <description> element
	  ▪ Must be bound within a <langstring> element
	• Maximum of 2048 characters

Req No	Requirement
2.3.1-8.10	The Meta-data XML document shall contain a **<description>** element (LOM element 5.10) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <description> • Bound within 1 or more <langstring> elements • Maximum of 1024 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <description> • Bound within 1 or more <langstring> elements • Maximum of 1024 characters
2.3.1-8.11	The Meta-data XML document shall contain a **<language>** element (LOM element 5.11) that adheres to the following: Raw Media (Asset) • 0 or more element tags <language> • Up to 8 supported – Not a conformance check, warning only • Maximum of 128 characters • Value must adhere to the ISO 639[11] and ISO 3166[12] standards o LanguageID = Langcode('-'Subcode): en-GB ▪ Langcode – ISO 639[11] ▪ Subcode – ISO 3166[12] Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tags <language> • Up to 8 supported – Not a conformance check, warning only • Maximum of 128 characters • Value must adhere to the ISO 639[11] and ISO 3166[12] standards o LanguageID = Langcode('-'Subcode): en-GB ▪ Langcode – ISO 639[11] ▪ Subcode – ISO 3166[12]
2.3.1-9	The Meta-data XML document shall contain a **<rights>** element (LOM element 6) that adheres to the following: Raw Media (Asset) • 1 and only 1 element tag <rights> Content (SCO/ Block) / Course (Content Aggregation) • 1 and only 1 element tag <rights>
2.3.1-9.1	The Meta-data XML document shall contain a **<cost>** element (LOM element 6.1) that adheres to the following: Raw Media (Asset) • 1 and only 1 tag <cost> • Bound within 1 and only 1 <langstring> elements • Member of the following Restricted Vocabulary list o User_defined o See_classification

Req No	Requirement
	○ yes ○ no Content (SCO/ Block) / Course (Content Aggregation) • 1 and only 1 tag <cost> • Bound within 1 and only 1 <langstring> elements • Member of the following Restricted Vocabulary list ○ User_defined ○ See_classification ○ yes ○ no
2.3.1-9.2	The Meta-data XML document shall contain a **<copyrightandotherrestrictions>** element (LOM element 6.2) that adheres to the following: Raw Media (Asset) • 1 and only 1 tag <copyrightandotherrestrictions> • Bound within 1 and only 1 <langstring> elements • Member of the following Restricted Vocabulary list ○ User_defined ○ See_classification ○ yes ○ no Content (SCO/ Block) / Course (Content Aggregation) • 1 and only 1 tag <copyrightandotherrestrictions> • Bound within 1 and only 1 <langstring> elements • Member of the following Restricted Vocabulary list ○ User_defined ○ See_classification ○ yes ○ no
2.3.1-9.3	The Meta-data XML document shall contain a **<description>** element (LOM element 6.3) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <description> • Bound within 1 or more <langstring> elements • Maximum of 1024 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <description> • Bound within 1 or more <langstring> elements • Maximum of 1024 characters

Req No	Requirement
2.3.1-10	The Meta-data XML document shall contain a **<relation>** element (LOM element 7) that adheres to the following: Raw Media (Asset) • 0 or more element tags <relation> • Up to 32 supported – Not a conformance check, warning only Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tags <relation> • Up to 32 supported – Not a conformance check, warning only
2.3.1-10.1	The Meta-data XML document shall contain a **<kind>** element (LOM element 7.1) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <kind> • Bound within 1 and only 1 <langstring> element • Checked against the Best Practice Vocabulary list o User_defined o See_classification o IsPartOf o HasPart o IsVersionOf o HasVersion o IsFormatOf o HasFormat o References o IsReferencedBy o IsBasedOn o IsBasisFor o Requires o IsRequiredBy • If not a member of the Best Practice Vocabulary list, Maximum of 1024 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <kind> • Bound within 1 and only 1 <langstring> element • Checked against the Best Practice Vocabulary list o User_defined o See_classification o IsPartOf o HasPart o IsVersionOf o HasVersion o IsFormatOf o HasFormat o References

Req No	Requirement
	○ IsReferencedBy ○ IsBasedOn ○ IsBasisFor ○ Requires ○ IsRequiredBy • If not a member of the Best Practice Vocabulary list, Maximum of 1024 characters
2.3.1-10.2	The Meta-data XML document shall contain a **<resource>** element (LOM element 7.2) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <resource> Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <resource>
2.3.1-10.2.1	• The Meta-data XML document shall **NOT** contain an **<identifier>** element (LOM element 7.2.1). This element is **RESERVED** and is therefore, not present in the IMS Learning Resource Meta-data DTD[10].
2.3.1-10.2.2	The Meta-data XML document shall contain a **<description>** element (LOM element 7.2.2) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <description> • Bound within 1 or more <langstring> elements • Maximum of 1024 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <description> • Bound within 1 or more <langstring> elements • Maximum of 1024 characters
2.3.1-11	The Meta-data XML document shall contain a **<annotation>** element (LOM element 8) that adheres to the following: Raw Media (Asset) • 0 or more element tags <annotation> • Up to 32 supported – Not a conformance check, warning only Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tags <annotation> • Up to 32 supported – Not a conformance check, warning only
2.3.1-11.1	The Meta-data XML document shall contain a **<centity>** element (LOM element 8.1) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <centity> • Maximum of 1024 characters • Bound within a <vcard> element Content (SCO/ Block) / Course (Content Aggregation)

Req No	Requirement
	• 0 or 1 element tag <centity> • Maximum of 1024 characters • Bound within a <vcard> element

2.3.1-11.2	The Meta-data XML document shall contain a **<date>** element (LOM element 8.2) that adheres to the following:
	Raw Media (Asset)
	• 0 or 1 element tag <date>
	• Must be bound in a Date Type
	  ○ <datetime> element must be expressed as per ISO 8601[14]
	  ○ <datetime> element best practice: Universal date/time format
	  ○ <description> element
	  ▪ Must be bound within a <langstring> element
	  ▪ Maximum of 2048 characters
	Content (SCO/ Block) / Course (Content Aggregation)
	• 0 or 1 element tag <date>
	• Must be bound in a Date Type
	  ○ <datetime> element must be expressed as per ISO 8601[14]
	  ○ <datetime> element best practice: Universal date/time format
	  ○ <description> element
	  ▪ Must be bound within a <langstring> element
	  ▪ Maximum of 2048 characters

2.3.1-11.3	The Meta-data XML document shall contain a **<description>** element (LOM element 8.3) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <description> • Bound within 1 or more <langstring> elements • Maximum of 1024 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <description> • Bound within 1 or more <langstring> elements • Maximum of 1024 characters
2.3.1-12	The Meta-data XML document shall contain a **<classification>** element (LOM element 9) that adheres to the following: Raw Media (Asset) • 0 or more element tags <classification> • Up to 10 supported – Not a conformance check, warning only Content (SCO/ Block) / Course (Content Aggregation) • 1 or more element tags <classification> • Up to 10 supported – Not a conformance check, warning only

Req No	Requirement

2.3.1-12.1	The Meta-data XML document shall contain a **<purpose>** element (LOM element 9.1) that adheres to the following:
	Raw Media (Asset)
	• 0 or 1 element tag <purpose>
	• Bound within 1 and only 1 <langstring> element
	• Checked against the Best Practice Vocabulary list
	  ○ User_defined
	  ○ See_classification
	  ○ Discipline
	  ○ Idea
	  ○ Prerequisite
	  ○ Educational Objective
	  ○ Accessibility Restrictions
	  ○ Educational Level
	  ○ Skill Level
	  ○ Security Level
	• If not part of the Best Practice Vocabulary list, maximum of 128 characters
	Content (SCO/ Block) / Course (Content Aggregation)
	• 1 and only 1 element tag <purpose>
	• Bound within 1 and only 1 <langstring> element
	• Checked against the Best Practice Vocabulary list
	  ○ User_defined
	  ○ See_classification
	  ○ Discipline
	  ○ Idea
	  ○ Prerequisite
	  ○ Educational Objective
	  ○ Accessibility Restrictions
	  ○ Educational Level
	  ○ Skill Level
	  ○ Security Level
	• If not part of the Best Practice Vocabulary list, maximum of 128 characters

2.3.1-12.2	The Meta-data XML document shall contain a **<taxonpath>** element (LOM element 9.2) that adheres to the following: Raw Media (Asset) • 0 or more element tags <taxonpath> • Up to 16 supported – Not a conformance check, warning only Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tags <taxonpath> • Up to 16 supported – Not a conformance check, warning only

Req No	Requirement
2.3.1-12.2.1	The Meta-data XML document shall contain a **<source>** element (LOM element 9.2.1) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <source> • Maximum of 1024 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <source> • Maximum of 1024 characters
2.3.1-12.2.2	The Meta-data XML document shall contain a **<taxon>** element (LOM element 9.2.2) that adheres to the following: Raw Media (Asset) • 0 or more element tags <taxon> • Up to 16 supported – Not a conformance check, warning only Content (SCO/ Block) / Course (Content Aggregation) • 0 or more element tags <taxon> • Up to 16 supported – Not a conformance check, warning only
2.3.1-12.2.2.1	The Meta-data XML document shall contain a **<id>** element (LOM element 9.2.2.1) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <id> • Maximum of 128 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <id> • Maximum of 128 characters
2.3.1-12.2.2.2	The Meta-data XML document shall contain a **<entry>** element (LOM element 9.2.2.2) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <entry> • Bound within 1 or more <langstring> elements • Maximum of 512 characters Content (SCO/ Block) / Course (Content Aggregation) • 0 or 1 element tag <entry> • Bound within 1 or more <langstring> elements • Maximum of 512 characters
2.3.1-12.3	The Meta-data XML document shall contain a **<description>** element (LOM element 9.3) that adheres to the following: Raw Media (Asset) • 0 or 1 element tag <description> • Bound within 1 or more <langstring> elements • Maximum of 2048 characters Content (SCO/ Block) / Course (Content Aggregation) • 1 and only 1 element tag <description>

Req No	Requirement
	• Bound within 1 or more <langstring> elements • Maximum of 2048 characters
2.3.1-12.4	The Meta-data XML document shall contain a **<keywords>** element (LOM element 9.4) that adheres to the following: Raw Media (Asset) • 0 or more element tags <keywords> • Up to 8 supported – Not a conformance check, warning only • Bound within 1 or more <langstring> elements • Maximum of 1024 characters Content (SCO/ Block) / Course (Content Aggregation) • 1 or more element tags <keywords> • Up to 8 supported – Not a conformance check, warning only • Bound within 1 or more <langstring> elements Maximum of 1024 characters

Table 2.3.1b Course, Content and Raw Media Meta-Data Conformance Requirements

This page intentionally left blank.

3. SECTION IV (Page Number Style)

SECTION 3

Detailed Test Procedures

This page intentionally left blank.

3.1. Conformance Test Suite Overview

These procedures require the use of the ADL SCORM Version 1.1 Conformance Test Suite Version 1.1 (referred to hereafter as "the Test Suite").

The Test Suite has been developed to help developers gauge the conformance of their software tools and learning content against the SCORM Version 1.1. In addition, the Test Suite is used as the basis for determining conformance of LMSs and learning content for ADL certification within the categories described in the Conformance Matrix in Section 1.

The following procedures assume that the Test Suite is installed in the **c:\ADL\TestSuite1_1_1ST** folder. If you have installed the Test Suite to a different location, please note this accordingly as you proceed through the following instructions.

3.2. LMS Test Procedures

3.2.1. About the LMS Conformance Test Software

The LMS Run-Time Environment Conformance Test software consists of two parts:

1. The LMS Run-Time Environment Test software that provides instructions for conducting the test and displays the test results.
2. Two Test Courses that are to be imported/loaded into the test subject LMS and experienced by fictitious students.

About the LMS Run-Time Environment Conformance Test Software

The LMS Runtime Environment Conformance Test consists of a user interface that provides step-by-step instructions for conducting the test as well as a log of each step performed by the testing software and their respective outcomes.

About the Test Courses

The Test Suite consists of two (2) test courses:

- ADL SCORM Test Course I
- ADL SCORM Test Course II

Both of these courses are intended to be imported or loaded into the test subject LMS and experienced by a mixture of fictitious students through the test subject LMS. Each Course and all of its contents reside in a sub-folder of the "courses" folder (**`c:\ADL\TestSuite1_1_1ST\lmsrte\courses`**). A CSF document that describes the internal hierarchy of each of the courses' SCOs also exists within the courses' folders.

The course contents are located as follows:

- ADL SCORM Test Course 01 located in folder:
  `c:\ADL\TestSuite1_1_1ST\lmsrte\courses\LMSTestcourse01`
- ADL SCORM Test Course 02 located in folder:
  `c:\ADL\TestSuite1_1_1ST\lmsrte\courses\LMSTestcourse02`

The CSF documents can be viewed using an XML editor or a text editor such as Windows Notepad. Understanding the hierarchy and order of the SCOs is critical for correctly configuring the test courses within the test subject LMS in the absence of automatically importing the courses using their respective CSF documents.

ADL SCORM Test Course I is comprised of seven (7) Sharable Content Objects (SCOs) that exercise the conformance requirements stated in Section 2.1. ADL SCORM Test Course II is comprised of three (3) SCOs. The purpose of this course is to exercise the LMS's ability to support the various vocabulary items associated with the cmi.core.credit and cmi.core.lesson_mode data model elements.

During execution of the test SCOs from within the LMS, if at any time a mandatory conformance requirement is found to be not implemented according to the conformance requirements, the test SCOs contained in this course will attempt to proceed with the test to completion. This may not be possible in all cases, depending on the nature of the failure. For this reason, when performing self-testing, it may be necessary to re-run the test multiple times after one failure is corrected, until all failures have been discovered and corrected. (Note that there may be restrictions on running the test multiple times when undergoing audited certification testing). The test log that is provided in the right frame of the test software will provide as much information as is feasible to describe the failure.

Also note that the LMS is responsible for controlling the sequencing and navigation between SCOs. The SCORM Version 1.1 does not specify the requirements for LMS implementation of this responsibility. The test SCOs that comprise the ADL Test Course must be launched sequentially. For example, the test subject LMS may provide a menu of hyperlinks that correspond to the test SCOs, or it may provide "back and next" buttons to allow sequential navigation between SCOs, or it may automatically launch SCOs in sequence. The test subject LMS may or may not control or restrict sequencing and navigation based on the CSF.

Each SCO is expected to be launched in sequential order using the LMS. SCOs may be launched by the LMS automatically if the LMS supports automatic (e.g. adaptive) sequencing, or manually via a menu or similar mechanism provided by the LMS. Upon launch, each SCO locates the LMS's API implementation and makes a series of API calls

to the LMS. The purpose of the test course SCOs is to audit the correctness and completeness of the LMS's Run-time Environment implementation.

3.2.2. LMS Conformance Test Procedures

Purpose:

The purpose of this test is to determine the conformance of a test subject LMS to the requirements for "LMS Run-Time Environment Conformance" within the three categories defined in the conformance matrix:

- LMS Run-Time Environment Conformant – Minimum (LMS-RTE1)
- LMS Run-Time Environment Conformant – Minimum with Some Optional Data Model Elements (LMS-RTE2)
- LMS Run-Time Environment Conformant – Minimum with All Optional Data Model Elements (LMS-RTE3)

Prerequisites:

The LMS Run-Time Environment Test requires Microsoft Internet Explorer 5.x. Note that none of the tests contained in the Test Suite have been tested with Microsoft Internet Explorer 6.x or Netscape 6.x.

The hardware configuration for the test subject LMS is not specified. At least three potential server configurations are envisioned, any of which may be employed for the test:

1. Test courses installed on remote vendor hardware.
2. Test courses installed on local vendor provided hardware.
3. Test courses installed on local, auditor provided hardware.

The test subject LMS can be accessed using the web browser either locally (on the same machine that the LMS server is located on, or remotely, using a different client workstation.

The Test Suite must be installed on the client machine, whether this is the same machine that hosts the LMS server, or a remote workstation.

Other hardware configurations are allowable at the discretion of the auditor, after consultation with the ADL Certification Council.

See the Test Suite readme.htm file for more information regarding the installation and use of the Test Suite.

Procedures:

The following are the detailed test procedure steps for conducting conformance testing of an LMS toward certification within the three categories as described in the Conformance Matrix in Section 1:

Step No	Instructions
3.2.2-1	Install the test subject LMS according to the vendor's instructions (or with the vendor's assistance) and assure that all software components, servers, services, etc. are running.
3.2.2-2	Install the ADL SCORM Version 1.1 Conformance Test Suite Version 1.1 (the Test Suite) according to the installation instructions provided with the software. The Test Suite is available for download from www.adlnet.org. The remainder of the procedure steps assumes that the Test Suite client was installed to the `c:\adl\TestSuite1_1_1ST` directory. If you installed the software to a different location, modify the following instructions accordingly.
3.2.2-3	Run the Test Suite on the same client workstation that the LMS will be launched from by opening the file `c:\ADL\TestSuite1_1_1ST\main.htm` using the web browser.
3.2.2-4	Click the LMS Run-Time Environment Conformance Test hyperlink. This will display the LMS Conformance Test window. The instructions provided in the left-hand frame reflect the remainder of the procedure steps in this table. Note that as the test is executed, a log of the test results is produced dynamically in the right hand frame.
3.2.2-5	Enter the information about the test subject LMS (product name, version, and vendor/developer name) in the fields provided and click the "Continue" button when finished.
3.2.2-6	Assure that the LMS Test Courses do not exist in the LMS. Manually inspect the LMS course catalog, or list of available courses to assure that the ADL SCORM Test Course I, and/or ADL SCORM Test Course II have not been previously loaded or imported. If the test courses are present within the LMS, remove them.
3.2.2-7	Configure ADL SCORM Test Course I in the LMS.

Step No	Instructions
	Determine if the vendor claims that the LMS supports import of courses using SCORM Version 1.1 CSF XML documents.
3.2.2-7.1	If the vendor claims that the LMS **does support** import of courses using SCORM Version 1.1 CSF XML documents, then attempt to import the ADL SCORM Test Course I using the CSF XML document: **`c:\ADL\TestSuite1_1_1ST\lmsrte\courses\CSF-LMSTestCourse01.xml`** (Note that it is permissible to manually copy the courses' directories and/or contents to the LMS's web server location if needed. The purpose of this step is to determine if the LMS supports the import of data about the course using the CSF, for which no formal conformance requirements exist.)
3.2.2-7.1.1	Observe the CSF import process for the course as it executes to determine if the LMS appears to be able to successfully import the course using the CSF.
3.2.2-7.1.2	If the LMS generates any error messages to indicate that the import process failed for the test course, note these error messages along with any additional relevant information using the "Add Comments" hyperlink in the log window. Click the "Course Import was attempted but was not successful" radio button, then click continue. The test is concluded. Proceed to step 3.2.2-14.
3.2.2-7.1.3	If the LMS appears to import the course successfully, attempt to manually validate that the course was imported successfully by inspecting the LMS's course listing/catalog, etc. If it appears that the import was successful, then click the radio button for: "The LMS was able to successfully import the Course using the supplied CSF Document." Then click "Continue".
3.2.2-7.2	If the vendor claims that the LMS **does not support** the ability to import courses using SCORM Version 1.1 CSF XML documents, or the course could not be imported using the CSF in step 3.2.2-7.1, then manually import ADL SCORM Test Course I. A CSF document that describes the internal hierarchy of the course's SCOs exists within the courses' folders. The CSF documents can be viewed using an XML editor or a text editor such

Step No	Instructions
	as Microsoft Windows Notepad. Understanding the hierarchy and order of the SCOs is critical for correctly configuring the test course within the test subject LMS in the absence of automatically importing the courses using its CSF. The CSF for ADL SCORM Test Course I is located in: **c:\ADL\TestSuite1_1_1ST\lmsrte\courses\CSF-LMSTestCourse01.xml** The contents of the course are located in: **c:\ADL\TestSuite1_1_1ST\lmsrte\courses\LMSTestcourse01**
3.2.2-7.2.1	If the course can be configured manually and the LMS appears to have had the course manually imported successfully, then attempt to manually validate that the course was imported successfully by inspecting the LMS's course listing/catalog, etc. If it appears that the import was successful, then click the radio button for: "The LMS does not support Course Interchange via CSF XML Documents. The course was loaded into the LMS manually." Then click the "Continue" button.
3.2.2-7.2.2	If it was not possible to import the course manually, or by importing the supplied CSF automatically, then click the radio button for: "The course could not be automatically or manually imported into the LMS". Then click the "Continue" button. The test is concluded.
3.2.2-8	Configure ADL SCORM Test Course II in the LMS. Based on the same determination that was made in Step 3.2.2-7, regarding the vendor's claims that the LMS does or does not support import of courses using SCORM Version 1.1 CSF XML documents:
3.2.2-8.1	If the vendor claims that the LMS **does support** import of courses using SCORM Version 1.1 CSF XML documents, then attempt to import the ADL SCORM Test Course II using the CSF XML document: **c:\ADL\TestSuite1_1_1ST\lmsrte\courses\CSF-**

Step No	Instructions
	`LMSTestCourse02.xml` (Note that it is permissible to manually copy the courses' directories and/or contents to the LMS's web server location if needed. The purpose of this step is to determine if the LMS supports the import of data about the course using the CSF, for which no formal conformance requirements exist.)
3.2.2-8.1.1	Observe the CSF import process for the course as it executes to determine if the LMS appears to be able to successfully import the course using the CSF.
3.2.2-8.1.2	If the LMS generates any error messages to indicate that the import process failed for the test course, note these error messages along with any additional relevant information using the "Add Comments" hyperlink in the log window. Click the "Course Import was attempted but was not successful" radio button, then click continue. The test is concluded. Proceed to step 3.2.2-14.
3.2.2-8.1.3	If the LMS appears to import the course successfully, attempt to manually validate that the course was imported successfully by inspecting the LMS's course listing/catalog, etc. If it appears that the import was successful, then click the radio button for: "The LMS was able to successfully import the Course using the supplied CSF Document." Then click the "Continue" button.
3.2.2-8.2	If the vendor claims that the LMS **does not support** the ability to import courses using SCORM Version 1.1 CSF XML documents, or the course could not be imported using the CSF in step 3.2.2-8.1, then manually import ADL Test Course II. A CSF document that describes the internal hierarchy of the course's SCOs exists within the courses' folders. The CSF documents can be viewed using an XML editor or a text editor such as Windows Notepad. Understanding the hierarchy and order of the SCOs is critical for correctly configuring the test course within the test subject LMS in the absence of automatically importing the courses using its CSF. The CSF for ADL SCORM Test Course II is located in: **`c:\ADL\TestSuite1 1 1ST\lmsrte\courses\CSF-`**

Step No	Instructions
	`LMSTestCourse02.xml` The contents of the course are located in: **`c:\ADL\TestSuite1_1_1ST\lmsrte\courses\LMSTestcourse02`**
3.2.2-8.2.1	If the course can be configured manually and LMS appears to have had the course manually imported successfully, then attempt to manually validate that the course was imported successfully by inspecting the LMS's course listing/catalog, etc. If it appears that the import was successful, then click the radio button for: "The LMS does not support Course Interchange via CSF XML Documents. The course was loaded into the LMS manually." Then click the "Continue" button.
3.2.2-8.2.2	If it was not possible to import the course manually, or by importing the supplied CSF automatically, then click the radio button for: "The course could not be automatically or manually imported into the LMS". Then click the "Continue" button. The test is concluded.
3.2.2-9	Enroll a student for ADL SCORM Test Course I as follows: • Last Name: Student • First Name: Joe The student should be registered for the course for "credit" (i.e. the LMS should return a value of "credit" for the cmi.core.credit data model element for all SCOs in this course for this student.) If the vendor claims that the LMS supports the cmi.core.lesson_mode data model element and that the LMS supports the "mode" feature (i.e. cmi.core.lesson_mode is supported by allowing a student to take a course, or a portion of a course in a "browse" or "review" mode), then assure that the student is configured such that the lesson mode for the entire course (all SCOs) is "normal" (i.e. the LMS should return a value of "normal" for the cmi.core.lesson_mode data model element for all SCOs in this course for this student.) Otherwise, proceed with registering the

Step No	Instructions
	student without regard for the cmi.core.lesson_mode value. The presumption is that the activity to "Register" a student for a course includes defining a user login Id and perhaps a password. Record the student's login Id and password for future use.
3.2.2-9.1	The test will prompt the operator as follows: "Did the LMS allow you to successfully enroll the student for the Test Course?" If any problems are encountered that prevent registering the student for ADL SCORM Test Course I, note any error messages along with any additional information using the "Add Comments" hyperlink in the log window. Click the "No" button. The test is concluded. Proceed to step 3.2.2-14. If the student is successfully enrolled in ADL SCORM Test Course I, click the yes radio button, and then click the "Continue" button.
3.2.2-9.2	The test instructions will prompt the operator to enter the student Id of the enrolled student as assigned by the LMS. If the student Id assigned by the LMS is visible, record it in the space provided in the test instructions. (This should be the value that the LMS will return for the cmi.core.student_id data model element). Click the "Complete" button.
3.2.2-10	Enroll a student in ADL SCORM Test Course II.
3.2.2-10.1	The operator is asked to determine if the LMS supports the ability to configure a student such that the student can take a given course for either "credit" or "no-credit" (i.e. supports both vocabulary elements for the cmi.core.credit data model element).
3.2.2-10.1.1	If the vendor claims that the LMS **does support** the ability to configure a student such that the student can take the course for either "credit" or "no-credit", then click the "Yes" radio button.

Step No	Instructions
3.2.2-10.1.2	If the vendor claims that the LMS **does not support** the ability to configure a student such that the student can take the course for either "credit" or "no-credit", then click the "No" radio button.
3.2.2-10.2	The operator is asked to determine if the LMS supports the ability to configure a student such that the student can take a given course, and/or a portion of a course in "browse" mode or "normal" mode, i.e. the LMS supports the cmi.core.lesson_mode data model element and in addition supports the "browse", "normal" and "review" vocabularies for the element. Note that if the LMS does not support the cmi.core.lesson_mode element, a mode of "normal" is to be assumed by each SCO. Further, it is possible that the LMS may support the cmi.core.lesson_mode data model element, but not properly support the mode functionality as intended by the SCORM.
3.2.2-10.2.1	If the vendor claims that the LMS **does support** the ability to configure a student such that the student can take a course, and/or a portion of a course in "browse" or "normal" mode, then click the "Yes" radio button and then click the "Continue" button.
3.2.2-10.2.2	If the vendor claims that the LMS **does not support** the ability to configure a student such that the student can take a course, and/or a portion of a course in "browse" or "normal" mode, then click the "No" radio button and then click the "Continue" button.
3.2.2-10.3	Enroll a student for ADL SCORM Test Course II as follows: • Last Name: Learner • First Name: Mary If the operator selected "Yes" in step 3.2.2.10.1, then the operator will be instructed to register the student for the course for "no-credit" (i.e. the LMS should return a value of "no-credit" for the cmi.core.credit data model element for all SCOs in this course for this student). If the operator selected "No" in step 3.2.2.10.1, then the operator will be instructed to register the student for the course for "credit" (i.e. the LMS should return a value of "credit" for the cmi.core.credit data model element for all SCOs in this course for

Step No	Instructions
	this student). If the operator selected "Yes" in step 3.2.2.10.2, then the operator will be instructed to assure that the student is configured such that the lesson mode for the entire course (all SCOs) is "browse" (i.e. the LMS should return a value of "browse" for the cmi.core.lesson_mode data model element for all SCOs in this course for this student). If the operator selected "No" in step 3.2.2.10.2, then it is assumed that the LMS either does not implement the cmi.core.lesson_mode element, or that the student is configured such that the lesson mode for the entire course (all SCOs) is "normal" (i.e. the LMS should return a value of "normal" for the cmi.core.lesson_mode data model element for all SCOs in this course for this student). The presumption is that the activity to "Register" a student for a course includes defining a user login Id and perhaps a password. Record the student's login Id and password for future use.
3.2.2-10.3.1	The test will prompt the operator as follows: "Did the LMS allow you to successfully enroll the student for the Test Course?" If any problems are encountered that prevent registering the student for ADL SCORM Test Course II, note any error messages along with any additional information using the "Add Comments" hyperlink in the log window. Click the "No" button. The test is concluded. Proceed to step 3.2.2-14. If the student is successfully enrolled in ADL SCORM Test Course II, click the yes radio button, and then click the "Continue" button.
3.2.2-10.3.2	The test instructions will prompt the operator to enter the student Id of the enrolled student as assigned by the LMS. If the student Id assigned by the LMS is visible, record it in the space provided in the test instructions. (This should be the value that the LMS will return for the cmi.core.student_id data model element.) Click the "Complete" button.

Step No	Instructions
3.2.2-11	Take ADL Test Course I as Joe Student Open a new instance of the web browser and access the test subject LMS. Login as the fictitious student, Joe Student that was created/registered in Step 3.2.2-9.
3.2.2-11.1	If you are unable to access the LMS, or unable to log into the LMS note any error messages along with any additional information using the "Add Comments" hyperlink in the log window. Click the "Abort Test Now" button. The test is concluded. Proceed to step 3.2.2-14.
3.2.2-11.2	If you are able to successfully access and log into the LMS using the fictitious student login information, begin taking the ADL SCORM Test Course I as student "Joe Student". Each SCO in the test course must be launched sequentially (SCO01 through SCO07). The SCORM Version 1.1 does not provide specific guidance for sequencing and navigation between SCOs. The LMS may automatically launch SCOs in sequence, or may provide a means for the user to navigate between SCOs. Manually launch each of the seven SCOs, if needed, or wait for the LMS to launch each SCO if automatic sequencing/navigation is supported.
3.2.2-11.3	If the SCO is not launched successfully (e.g. if another SCO is launched instead, or no SCO is launched), note any error messages along with any additional information using the "Add Comments" hyperlink in the log window. The test is concluded. Proceed to step 3.2.2-14.
3.2.2-11.4	If the SCO is successfully launched, (e.g. the SCO is displayed in the LMS designated content window), the SCO will display an overview of the conformance requirements that are tested by the SCO. The SCO will commence testing and will display a status of "Testing in Progress…" When the SCO completes its testing, it will display a status of "This SCO testing completed." Note that SCO04 will ask the operator to exit the LMS (log out) and re-enter prior to SCO04 being complete. Also note that SCO07 is the "ad-hoc" testing SCO. This SCO

Step No	Instructions
	provides the operator with the opportunity to exercise SCORM Run-Time Environment interaction with the test subject LMS beyond that of the scripted SCOs. The operator must initiate successful calls to LMSInitialize and LMSFinish at some point during the execution of this SCO in order to allow ADL SCORM Test Course I to complete. The operator may exercise API Adapter function calls in any order desired. The results of the function calls are displayed in SCO user interface as well as in the test log. When testing has been completed, to the satisfaction of the Test Suite operator, the Course may be exited. Note that causing the ad-hoc SCO to call the LMSFinish function correctly may cause the course to end automatically, otherwise the operator may quit or exit in any manner supported by the LMS. As each SCO executes its tests, messages describing the tests and their respective outcomes are written to the test login the Test Suite's right frame. In the event of a severe failure, the test may abort. If the test aborts, proceed to step 3.2.2-14.
3.2.2-11.5	When all of the SCOs for ADL SCORM Test Course I have been completed, return to the test instructions frame and click the "Course I Complete" button. Note that the "Abort Test Now" button may be used at any time during ADL SCORM Course I to abort the test.
3.2.2-12	Take ADL Test Course II as Mary Learner Open a new instance of the web browser and access the test subject LMS. Login as the fictitious student, Mary Learner that was created/registered in Step 3.2.2-10.
3.2.2-12.1	If you are unable to access the LMS, or unable to log into the LMS note any error messages along with any additional information using the "Add Comments" hyperlink in the log window. Click the "Abort Test Now" button. The test is concluded. Proceed to step 3.2.2-14.
3.2.2-12.2	If you are able to successfully access and log into the LMS using the fictitious student login information, begin taking the ADL SCORM Test Course II as student "Mary Learner". Each SCO in the test course must be launched sequentially (SCO01 through SCO03). The SCORM Version 1.1 does not provide specific guidance for sequencing and navigation between SCOs. The LMS may automatically launch SCOs in sequence, or may

Step No	Instructions
	provide a means for the user to navigate between SCOs. Manually launch each of the three SCOs, if needed, or wait for the LMS to launch each SCO if automatic sequencing/navigation is supported.
3.2.2-12.3	If the SCO is not launched successfully (e.g. if another SCO is launched instead, or no SCO is launched), then note any error messages along with any additional information using the "Add Comments" hyperlink in the log window. The test is concluded. Proceed to step 3.2.2-14.
3.2.2-12.4	If the SCO is successfully launched, (e.g. it is displayed in the LMS designated content window), the SCO will display an overview of the conformance requirements that are tested by the SCO. The SCO will commence testing and will display a status of "Testing in Progress…" When the SCO completes its testing, it will display a status of "This SCO testing completed." Note that SCO02 will ask the operator to exit the LMS (log out) and re-enter prior to SCO02 being complete. Also note that SCO03 is the "ad-hoc" testing SCO. This SCO provides the operator with the opportunity to exercise SCORM Run-Time Environment interaction with the test subject LMS beyond that of the scripted SCOs. The operator must initiate successful calls to LMSInitialize and LMSFinish at some point during the execution of this SCO in order to allow ADL SCORM Test Course II to complete. The operator may exercise API Adapter function calls in any order desired. The results of the function calls are displayed in SCO user interface as well as in the test log. When testing has been completed, to the satisfaction of the Test Suite operator, the Course may be exited. Note that causing the ad-hoc SCO to call the LMSFinish function correctly may cause the course to end automatically, otherwise the operator may quit or exit in any manner supported by the LMS. As each SCO executes its tests, messages describing the tests and their respective outcomes are written to the test log in the Test Suite's right frame. In the event of a severe failure, the test may abort. If the test aborts, proceed to step 3.2.2-14.

Step No	Instructions
3.2.2-12.5	When all of the SCOs for ADL SCORM Test Course II have been completed, return to the test instructions frame and click the "Course II Complete" button. Note that the "Abort Test Now" button may be used at any time during ADL SCORM Course II to abort the test.
3.2.2-13	View the log to determine the results of the test. If the test completes successfully, the log will display an indication of the conformance category achieved by the test subject. Note that comments can be added to the log at any time using the "Add Comments" hyperlink in the log frame.
3.2.2-14	To save the log, click on "Save Log" and choose the destination file name.
3.2.2-15	To print the log, click on "Print Log."
3.2.2-16	The test is concluded.

3.3. Sharable Content Object Run-Time Test Procedures

3.3.1. About the SCO Conformance Test Software

The SCO Run-Time Environment Conformance Test software simulates a "SCORM Version 1.1 Conformant – Minimum with All Optional Data Model Elements" (LMS-RTE3) LMS. The SCO Test software launches one or more test subject SCOs. The SCO Test software provides an API Adapter, which the SCO must find in order to invoke API Adapter function calls. The SCO Test software responds appropriately to each API Adapter function call initiated by the test subject SCO.

3.3.2. SCO Conformance Test Procedures

Purpose:

The purpose of this test is to determine the conformance of a test subject SCO to the requirements for "SCO Run-Time Environment Conformance" within the four categories defined in the conformance matrix:

- SCORM Version 1.1 Run-Time Environment Conformant – Minimum (SCO-RTE1)
- SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Mandatory Data Model Elements (SCO-RTE1+Mandatory)
- SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Optional Data Model Elements (SCO-RTE1+Optional)
- SCORM Version 1.1 Run-Time Environment Conformant – Minimum with Some Optional and Some Mandatory Data Model Elements (SCO-RTE1+Mandatory+Optional)

Prerequisites:

The SCO Run-Time Environment Test is capable of running in the following web browsers:

- Microsoft Internet Explorer 5.x
- Netscape 4.7

Note that none of the tests contained in the Test Suite, including the SCO Run-Time Environment Test have been tested with Microsoft Internet Explorer 6.x or Netscape 6.x.

The test subject SCO can be accessed from the local file system, or via http from a remote or local web server using a Universal Resource Identifier (URI).

The Test Suite must be installed on a local client machine, whether this is the same machine that hosts the SCO's web server, or not.

See the Test Suite installation instructions for more information regarding the installation and use of the Test Suite. It is recommended to use Internet Explorer as the browser to run the Test Suite. However, if the SCOs being tested reside in a different domain as the installation location of the Test Suite, then the Netscape Navigator browser should be used.

Procedures:

The following are the detailed test procedure steps for conducting conformance testing of a SCO toward certification within the four categories as described in the Conformance Matrix in Section 1:

Step No	Instructions
3.3.2-1	Install the ADL SCORM Version 1.1 Conformance Test Suite Version 1.1 (the Test Suite) according to the installation instructions provided with the software. The Test Suite is available for download from www.adlnet.org. The remainder of the procedure steps assumes that the Test Suite client was installed to the `c:\ADL\TestSuite1_1_1ST` directory. If you installed the software to a different location, modify the following instructions accordingly.
3.3.2-2	Run the Test Suite by opening the file `c:\ADL\TestSuite1_1_1ST\main.htm` using Microsoft Internet Explorer Version 5.0.x or 5.5.x or Netscape Navigator 4.7.
3.3.2-3	Click the "SCO Run-Time Environment Conformance Test" hyperlink. This will display the SCO Conformance Test window. The instructions provided in the left-hand frame reflect the remainder of the procedure steps in this table. Note that as the test is executed, a log of the test results is displayed in the right hand frame.
3.3.2-4	Enter the information about the test subject SCO (SCO name, version, and vendor/developer name) in the fields.
3.3.2-5	Enter the "LMSInitialize Timeout Value". This is the number of seconds that the SCO Test Software should wait after a SCO is launched to receive a call to LMSInitialize before timing out.

Step No	Instructions
3.3.2-6	Select the “Single” or “Multiple” SCO Test option.
3.3.2-7	Select the SCO(s) to be tested.
3.3.2-7.1	If testing a single SCO, then press the "Browse" button to select the SCO that will be tested.
3.3.2-7.2	If testing multiple SCOs, then: 1. Press the "Browse" button to select the first SCO that will be tested 2. Click the "Add to List" button to add the SCO to the list of SCOs 3. Repeat steps 1 and 2 for all SCO's to be tested
3.3.2-8	Click the “Launch SCO(s)” button to begin the test.
3.3.2-9	Exercise the functionality of the SCO. Note that there is no uniform manner in which this procedure can be specified because it is anticipated that test subject SCOs will vary widely in terms of size, functionality and interaction with the LMS. The Test Suite operator should exercise the capabilities of the test subject SCO as thoroughly and completely as is practical.
3.3.2-10	Once the Test Suite's LMS receives a call to LMSFinish("") from the test subject SCO, the Test Suite operator has two options: 1. If testing a Single SCO, the Test Suite operator should press the “Complete Test” button. This will close out the SCO and finish the test. 2. If testing Multiple SCOs, the Test Suite operator should press the “Complete Test” button. This will close out the current SCO being tested and start the process of launching the next SCO.
3.3.2-11	View the log to determine the results of the test. If the test completes successfully, the log will display an indication of the conformance category achieved by the test subject. Note that comments can be added to the log at any time using the “Add Comments” hyperlink in the log frame.

Step No	Instructions
3.3.2-12	Save the log, click on “Save Log” and choose the destination file name.
3.3.2-13	To print the log, click on “Print Log.”
3.3.2-14	The test is concluded.

3.4. Meta-Data Test Procedures

3.4.1. About the Meta-Data Conformance Test Software

The Meta-data Conformance Test software parses a test subject Raw Media, Content or Course Meta-data XML document to determine if the document meets the conformance criteria defined in Section 2.3.

3.4.2. Meta-Data Conformance Test Procedures

Purpose:

The purpose of this test is to determine the conformance of a test subject XML Meta-data document to the requirements for "Meta-data Conformance" within the four categories defined in the conformance matrix:

- SCORM Version 1.1 Meta-data XML Conformant – Minimum (MD-XML1)
- SCORM Version 1.1 Meta-data XML Conformant – Minimum with Optional Elements (MD-XML1+Optional)
- SCORM Version 1.1 Meta-data XML Conformant – Minimum with Extensions (MD-XML1+Extensions)
- SCORM Version 1.1 Meta-data XML Conformant – Minimum with Optional Elements and Extensions (MD-XML1+Optional+Extensions)

Prerequisites:

The Meta-data Test requires Microsoft Internet Explorer 5.x. Note that none of the tests contained in the Test Suite have been tested with Microsoft Internet Explorer 6.x or Netscape 6.x.

The test subject Meta-data XML document can be accessed from the local file system.

See the Test Suite installation instructions for more information regarding the installation and use of the Test Suite.

Procedures:

The following are the detailed test procedure steps for conducting conformance testing of a meta-data document toward certification within the four categories as described in the Conformance Matrix in Section 1:

Step No	Instructions
3.4.2-1	Install the ADL SCORM Version 1.1 Conformance Test Suite Version 1.1 (the Test Suite) according to the installation instructions provided with the software. The Test Suite is available for download from www.adlnet.org. The remainder of the procedure steps assumes that the Test Suite client was installed to the `c:\adl\TestSuite1_1_1ST` directory. If you installed the software to a different location, modify the following instructions accordingly.
3.4.2-2	Run the Test Suite by opening the file `c:\ADL\TestSuite1_1_1ST\main.htm` using Microsoft Internet Explorer Version 5.1 or higher.
3.4.2-3	Click the "Meta-data Conformance Test" hyperlink. This will display the Meta-data Conformance Test window. The instructions provided in the left-hand frame reflect the remainder of the procedure steps in this table. Note that as the test is executed, a log of the test results is displayed in the right hand frame.
3.4.2-4	Fill in the Meta-data document Name, Meta-data document Version/Release Number, and the Vender/Developer Information and click the "Continue" button.
3.4.2-5	Click the radio button that corresponds to the type of meta-data document (Raw Media, Content, Course) that is to be tested.
3.4.2-6	Type the name of the file meta-data document to be tested, or click the "Browse" button to choose the file that contains the meta-data document to be tested.
3.4.2-7	Click the "Begin Test" button to conduct the test.
3.4.2-8	View the log to determine the results of the test. If the test completes successfully, the log will display an indication of the conformance category achieved by the test subject. Note that comments can be added to the log at any time using the "Add Comments" hyperlink in the log frame.

Step No	Instructions
3.4.2-9	Save the log, click on “Save Log” and choose the destination file name.
3.4.2-10	To print the log, click on “Print Log.”
3.4.2-11	The test is concluded.

A. Appendix A (Appendix Number Style)

APPENDIX A

Acronym List

This page intentionally left blank.

Acronym Listing

ADL	Advanced Distributed Learning
AICC	Aviation Industry CBT Committee
API	Application Program Interface
ASCII	American Standard Code for Information Interchange
AU	Assignable Unit
AWT	Abstract Window Toolkit
CBI	Computer-Based Instruction
CBT	Computer-Based Training
CDATA	Character Data
CMI	Computer Managed Instructions
CSF	Content Structure Format
DC	Dublin Core
DoD	Department of Defense
DOL	Department of Labor
DTD	Document Type Definition
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IDA	Institute for Defense Analyses
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
ITS	Intelligent Tutoring Systems
LMS	Learning Management System
LOM	Learning Objects Metadata
LTSC	Learning Technology Standards Committee
MIME	Multipurpose Internet Mail Extensions
NGB	National Guard Bureau
OSTP	Office of Science and Technology Policy
PCDATA	Parsable Character Data
SCO	Sharable Content Object
SCORM	Sharable Content Object Reference Model
URI	Universal Resource Identifier
URL	Universal Resource Locator
W3C	World Wide Web Consortium
WWW	World Wide Web
XML	eXtensible Markup Language

This page intentionally left blank.

B. Appendix B (Appendix Number Style)

APPENDIX B

References

This page intentionally left blank.

References

1. Advanced Distributed Learning Co-Laboratories. (www.adlnet.org)
2. Aviation Industry CBT (Computer-Based Training) Committee. (www.aicc.org)
3. Institute of Electrical and Electronic Engineers (IEEE) Learning Technology Standards Committee (LTSC). (ltsc.ieee.org)
4. Institute for Defense Analyses (IDA). (www.ida.org)
5. IMS Global Learning Consortium, Inc. (www.imsproject.org)
6. AICC/CMI CMI001 Guidelines for Interoperability Version 3.4. October 23, 2000. Includes AICC Course Structure Format and AICC CMI Data Model Available at: www.aicc.org.
7. Executive Order 13111: Using Technology to Improve Training Opportunities for Federal Government Employees.
8. IMS Content Packaging Specification Version 1.1. Available at: www.imsproject.org.
9. IEEE Information Technology - Learning Technology - Learning Objects Metadata LOM: Base Scheme – v3.5 (1999-07-15) and Explanatory Notes for LOM v3.5. As referenced by the IMS Learning Resource Meta-data Specification Version 1.1. Available at: ltsc.ieee.org.
10. IMS Learning Resource Meta-data Specification Version 1.1. Includes: IMS Learning Resource Meta-data Information Model, IMS Learning Resource Meta-data XML Binding Specification, and IMS Learning Resource Meta-data Best Practice and Implementation Guide Available at: www.imsproject.org.
11. ISO 639: This is an international standard for the representation of languages. Version 1 uses two-letter language codes, e.g. 'en' for English, 'fr' for French, 'nl' for Dutch, etc. These language codes are a basis for the IETF registry of language tags, as specified in RFC 1766: Tags for the identification of languages. Available at: www.iso.ch.
12. ISO 3166: This is an international standard for the representation of country names, e.g. 'BE' for Belgium, 'CA' for Canada, 'FR' for France, 'GB' for United Kingdom, 'US' for United States, etc. Available at: www.iso.ch.

13. vCard: This standard defines how contact details for people and organizations can be represented.
Available at: www.imc.org/pdi.

14. ISO 8601: This is an international standard that specifies numeric representations of date and time.
Available at: www.iso.ch.

15. Sharable Content Object Reference Model Version 1.1. Available at www.adlnet.org.

16. World Wide Web Consortium (W3C). www.w3c.org
Includes: Universal Resource Locator, Universal Resource Identifier, Extensible Markup Language Version 1.0, Document Object Model (DOM) Specification.

C. Appendix C (Appendix Number Style)

APPENDIX C

Revision History

This page intentionally left blank.

Revision History

Version	Comment
Version 1.1	This is the first public release of this document
Version 1.1.1	Changes to reflect updates to the Test Suite Version 1.1.1 Multiplicity for langstrings holding vocabularies are now "1 and only 1." The data model vocabulary of "no credit" has been changed to "no-credit" Removed the "unique" vocabulary from the list of valid Interaction Types Test Suite software directory references have been changed from "TestSuite1_1" to "TestSuite1_1_1ST." Added clarification on requirements for setting cmi.core.session_time and the accumulation of cmi.core.total_time. Cosmetic clean up of test steps for the SCO Run-Time Environment Test. Clean up of steps dealing with completing of the test for both Single and Multiple SCO Testing. Added documentation stating the use case for running the SCO Run-Time Environment Test in Netscape Navigator. Changed the error code being compared against to "201" for function calls that pass in invalid arguments (LMSInitialize(), LMSFinish() and LMSCommit()). General cosmetic clean up.